KD11-K

PDP11/6X FP11E FP INST

MD-11-DQFPC-A

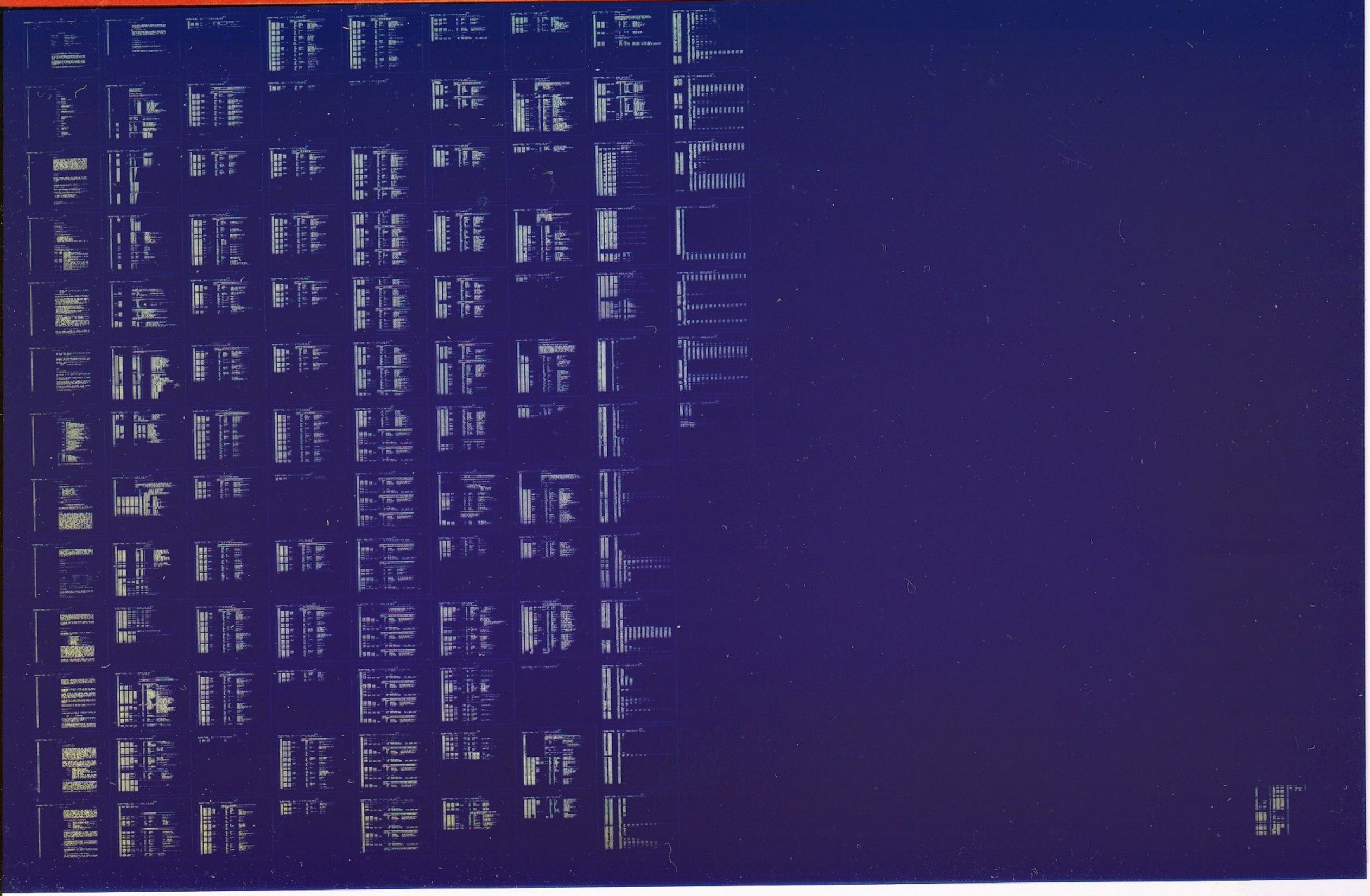
EP-DQFPC-A-DL-A

FICHE 1 OF 1

APR 1977

digital

MADE IN USA



```
000000                  .REPT   0
```

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DQFPC-A-D

PRODUCT NAME: PDP-11/6X - FP11-E FLOATING POINT UNIT INSTRUCTION EXERCISER

DATE: MARCH 1977

MAINTAINER: DIAGNOSTIC GROUP

AUTHOR: BOB BRAIN / STANLEY HARACKIEWICZ

REVISED BY: DONALD NORTH

CONTENTS
----

1. ABSTRACT

THIS PROGRAM CONTAINS THE PDP-11/6X FLOATING POINT PROCESSOR INSTRUCTION EXERCISER. THIS EXERCISER PROGRAM TESTS THE COMPLETE FLOATING POINT INSTRUCTION SET IN AN EXERCISER ENVIRONMENT, USING VARIOUS COMBINATIONS OF INSTRUCTIONS IN COMMON SOFTWARE CONFIGURATIONS AS THE BASIS FOR THE TESTS. EVERY CONCEIVABLE FLOATING POINT ERROR CONDITION IS DEVELOPED, AND THE RESPONSE CHECKED FOR CORRECTNESS. IN ADDITION, INTERACTION BETWEEN FLOATING POINT INSTRUCTION EXECUTION, INTERRUPTS, AND BASE MACHINE INTERRUPTS (USING THE DL11-W LINE AND/OR KW11-P PROGRAMMABLE CLOCKS) IS ALSO TESTED TO INSURE CORRECT PROCESSING BY BOTH THE BASE MACHINE AND FLOATING POINT MICROCODE. BOTH "HOT" (FP11-E OPTION) AND "WARM" (PDP-11/6X MICROCODE) FLOATING POINT UNITS CAN BE SELECTED FOR TESTING.

2. REQUIREMENTS

2.1 EQUIPMENT

PDP-11/6X STANDARD COMPUTER WITH MINIMUM 16K OF MEMORY. OPTIONAL FP11-E FLOATING POINT UNIT, IF SELECTED. THE DL11-W LINE CLOCK IS USED TO PROVIDE I/O INTERRUPTS DURING EXECUTION. IF PRESENT, THE KW11-P PROGRAMMABLE CLOCK WILL ALSO BE UTILIZED.

2.2 STORAGE

THE PROGRAM USES MEMORY 0-20452(8). THE UPPER 2.0K WORDS ARE RESERVED FOR THE XXDP MONITOR, IF EMPLOYED.

2.3 PRELIMINARY PROGRAMS

THE CPU, CACHE, AND MEMORY TEST PROGRAMS MUST BE RUN FIRST TO VERIFY THE CORRECT OPERATION OF THE BASE MACHINE.

THE PDP-11/6X - FP11-E FLOATING POINT PROCESSOR INSTRUCTION SET TESTS SHOULD THEN BE RUN IN THE FOLLOWING ORDER:

(1) DQFPA FPU BASIC INSTRUCTION TESTS
(2) DQFPB FPU ADVANCED INSTRUCTION TESTS
(3) DQFPC FPU INSTRUCTION EXERCISER
(4) DQFPD FPU ADD/SUB/MUL/DIV RANDOM EXERCISER

3. LOADING PROCEDURE

USE THE STANDARD PROCEDURE FOR ABSOLUTE TAPES, OR LOAD VIA XXDP MEDIA.

4. STARTING PROCEDURE

4.1 CONTROL SWITCH SETTINGS

SEE SECTION 5.1
SWITCH REGISTER (000000) IS WORST CASE TEST.

4.2 STARTING ADDRESS

THE PROGRAM MUST ALWAYS BE STARTED AT LOCATION 200(8).

4.3 PROGRAM/OPERATOR ACTION

LOADING VIA ABSOLUTE PAPERTAPE:

(1) LOAD PROGRAM INTO MEMORY USING ABS LOADER.
(2) LOAD ADDRESS 200 (8).
(3) SET SWITCHES (SEE SECTION 5.1)
SR=(000000) IS WORST CASE TEST.
(4) PRESS CONTROL/START TO BEGIN.
(5) PROGRAM TYPES IDENTIFICATION HEADER (VERIFY THAT THE CORRECT PROGRAM HAS BEEN LOADED!), AND EXECUTION BEGINS.

5. OPERATING PROCEDURE

5.1 OPERATIONAL SWITCH SETTINGS

THE DEFINITION OF THE SPECIFIC BITS IN THE SWITCH REGISTER (EITHER HARDWARE OR SOFTWARE) ARE AS FOLLOWS:

SW15=1	100000	HALT ON ERROR
SW14=1	040000	LOOP ON CURRENTLY EXECUTING TEST
SW13=1	020000	INHIBIT ERROR TYPEOUTS (WHICH IS AN "ERROR MESSAGE" RESULTING FROM AN ERROR DETECTED IN THE HARDWARE)
SW12=1	010000	INHIBIT STATUS TYPEOUTS (WHICH IS A NON-ERROR RELATED INFORMATIVE MESSAGE, SUCH AS "END PASS #XXX")
SW11=1	004000	INHIBIT ITERATIONS PER TEST
SW10	002000	SET=BELL ON ERROR/CLEAR=BELL ON PASS END
SW09=1	001000	LOOP ON ERROR
SW08=1	000400	LOOP ON TEST NUMBER IN "SLPTST" IF SET, THEN THE TEST SPECIFIED BY THE TEST NUMBER CONTAINED IN THE MEMORY WORD "SLPTST" (SEE PROGRAM LISTING) WILL SPECIFY THE DESIRED TEST ON WHICH TO LOOP.
SW01	000002	CLEAR=TEST HOT-FP/WARM-FP ALTERNATELY EACH PASS (IE, PASS#1 HFP, PASS#1 WFP, PASS#2 HFP, PASS#2 WFP, ETC) SET=TEST ONLY UNIT SPECIFIED IN SW00
SW00	000001	SET=SELECT WARM FP, IF SW01=1

CLEAR=SELECT HOT FP, IF SW01=1

NOTE FOR SW01, SW00 - IF NO HOT FP (FP11-E) IS PRESENT, THEN WARM FP (PDP-11/6X MICROCODE) IS AUTOMATICALLY SELECTED.

5.2 PROGRAM/OPERATOR ACTION

ONCE EXECUTION HAS BEGUN, MINIMAL OPERATOR INTERVENTION IS REQUIRED, UNLESS THE PROGRAM DETECTS AN ERROR IN THE HARDWARE.

IF ALL IS WELL, THE PROGRAM TYPES ITS NAME UPON BEGINNING; AND AT THE START OF EACH PASS, THE CURRENT PASS NUMBER (IN OCTAL) IS ECHOED. NOTE THAT SETTING SW<12>=1 WILL INHIBIT THE TYPEOUT OF THE BEGIN AND END PASS MESSAGES.
IF SW<10>=0, THE CONSOLE BELL WILL BE RUNG AT THE END OF EACH PASS. NOTE THAT ONLY SW<10> AFFECTS THE BELL RINGING AT END OF PASS - SW<12> HAS NO EFFECT ON THIS FUNCTION.

IF AN ERROR OCCURS DURING EXECUTION, MANY VARIATIONS IN ACTION ARE POSSIBLE DEPENDING UPON THE SWITCH SETTINGS.
SW<15>=1 WILL CAUSE THE CPU TO HALT AFTER AN ERROR.
SW<13>=1 WILL ALSO INHIBIT ANY ERROR MESSAGE TYPEOUT THAT WOULD OCCUR AT THIS TIME.
SW<10>=1 WILL CAUSE THE CONSOLE BELL TO BE RUNG ONLY WHEN AN ERROR IS DETECTED (AND NOT AT THE END OF A PASS).
SW<9>=1 CAUSES THE PROGRAM TO LOOP ON THE MOST RECENT ERROR, AS LONG AS IT CONTINUES TO OCCUR.

THERE ARE ALSO SEVERAL OTHER GENERAL USE FUNCTIONS DEFINED BY THE SWITCHES:
SW<11>=1 WILL INHIBIT THE ITERATIONS (=2000(10)) PERFORMED OF EACH TEST ON PASSES 2,3,4, ... THRU THE PROGRAM.
SW<14>=1 CAUSES THE PROGRAM TO LOOP INDEFINATELY ON THE CURRENTLY EXECUTING TEST.
SW<8>=1 CAUSES THE PROGRAM TO CONTINUE EXECUTION AS NORMAL, EXCEPT WHEN THE CONTENTS OF MEMORY WORD "$LPTST" MATCHES THE NUMBER OF THE TEST CURRENTLY EXECUTING. AT THIS POINT, THE TEST IS LOOPED ON INDEFINATELY, UNTIL EITHER SW<8>=0 OR "$LPTST" IS CHANGED. NOTE THAT IF "$LPTST" DOES NOT MATCH THE TEST NUMBER OF ANY TEST, THE CONTENTS OF "$LPTST" ARE EFFECTIVELY IGNORED, AND EXECUTION PROCEEDS NORMALLY.

5.3 HOT (FP11-E) / WARM (PDP-11/6X) SELECTION

WHEN THE PROGRAM IS STARTED (AT 200(8)), A MESSAGE IS OPTIONALLY PRINTED INDICATING THE PRESCENCE/ABSCENCE OF AN FP11-E HOT FLOATING POINT UNIT OPTION (BASED UPON WHETHER "WHAMI" BIT<04> IS 1/0 RESPECTIVELY).

IF NO FP11-E HOT FP OPTION IS PRESENT, THE MESSAGE IS TYPED,

AND ANY ATTEMPTS TO SELECT IT FOR TESTING VIA SW01 AND SW00 ARE IGNORED. ONLY WARM FP (PDP-11/6X MICROCODE) FLOATING POINT CAN BE TESTED/SELECTED.

IF THE FP11-E IS PRESENT, TEST SELECTION IS AS FOLLOWS:

WHEN SW01=0, THE HOT AND WARM FLOATING POINT UNITS ARE TESTED ALTERNATELY EACH PASS - IN THE ORDER (1) HOT, THEN (2) WARM. NOTE THAT EACH "PASS" NOW CONSISTS OF TWO SEPARATE SUB-PASSES.

WHEN SW01=1, THEN DEDICATED SELECTION OF A PARTICULAR UNIT IS SPECIFIED IN SW00:

SW00=0 --> TEST HFP FP11-E OPTION ONLY
SW00=1 --> TEST WFP PDP-11/6X MICROCODE ONLY

6. ERRORS

6.1 FORMAT OF MESSAGES

6.1.1 ALL ERROR MESSAGES CONSIST OF THREE LINES OF DATA:

THE FIRST LINE IS A BRIEF MESSAGE WHICH EXPLAINS WHAT ERROR WAS DETECTED (EG, THE RESULT OF THE "ABSF" INSTRUCTION WAS BAD).

THE PREFIX "HOT:" OR "WARM:" IS ALSO ATTACHED TO THE MESSAGE TO INDICATE THE SOURCE OF THE ERROR; THE FP11-E UNIT OR THE PDP-11/6X RESPECTIVELY.

THE SECOND LINE CONSISTS OF DATA HEADERS TO IDENTIFY THE VALUES TYPED OUT ON LINE THREE. THESE HEADERS WILL EITHER BE OF THE FORM "EXPECTED" AND "RECEIVED" DATA, OR WILL BE A MNEMONIC NAME OF A WORD LOCATION IN MEMORY OR REGISTERS.

THE THIRD LINE DISPLAYS THE CONTENTS OF THE LOCATIONS SPECIFIED BY LINE TWO AS SIX DIGIT OCTAL NUMBERS. NOTE THAT ALL DATA DISPLAYED IN ANY MESSAGES ARE OCTAL NUMBERS.

AS EXPLAINED IN SECTION 5.2, SETTING SW<13>=1 WILL SUPPRESS THE TYPING OF THESE MESSAGES.

6.1.2 FLOATING POINT UNIT DATA FORMATS:

FLOATING POINT STATUS WORD (FPS):

BIT##	OCTAL	FUNCTION
15	100000	FER - FLOATING ERROR FLAG SET WHEN EITHER FIUV, FIU, FIV, FIC ENABLED AND APPROPRIATE EXCEPTION OCCURRED.
14	040000	FID - FLOATING DISABLE INTERRUPTS NO FP INTERRUPTS TO VECTOR 244(8) IF SET.
13, 12		NOT USED
11	004000	FIUV - FLOATING UNDEFINED VARIABLE INTERRUPT IF SET, (-0) MEMORY DATA IS ERROR
10	002000	FIU - FLOATING INTR UNDERFLOW IF SET AND UNDERFLOW, SET FER, STORE ANSWER, EXPONENT WRONG BY +400(8) IF CLEAR AND UNDERFLOW, ANSWER <-- ZERO
9	001000	FIV - FLOATING OVERFLOW INTERRUPT IF SET AND OVERFLOW, SET FER, STORE ANSWER, EXPONENT WRONG BY -400(8) IF CLEAR AND OVERFLOW, ANSWER <-- ZERO
8	000400	FIC - FLOATING INTEGER CONVERSION INTERRUPT IF SET AND "STCFI" ERROR, ANSWER <-- ZERO, SET ERROR IF CLEAR AND "STCFI" ERROR, ANSWER <-- ZERO
7	000200	FD - FLOATING MODE 1=DOUBLE, 64 BIT OPERANDS (4W) 0=SINGLE, 32 BIT OPERANDS (2W)
6	000100	FL - INTEGER MODE 1=LONG, 32 BIT INTEGERS (2W) 0=SHORT, 16 BIT INTEGERS (1W)
5	000040	FT - ROUND/TRUNCATE MODE 1=TRUNCATE RESULTS 0=ROUND RESULTS
4	000020	FMM - PUT FP11-E ONLY IN MAINTENANCE MODE
3:0	000017	FN-FZ-FV-FC - FLOATING CONDITION CODES

FLOATING EXCEPTION CODES (FEC):

OCTAL	ENABLE	
00	(NONE)	(NOT USED)
02	(NONE)	FP OPCODE ERROR
04	(NONE)	FP DIVIDE-BY-ZERO ERROR
06	W/FIC	FP INTEGER CONVERSION ERROR
10	W/FIV	FP OVERFLOW ERROR
12	W/FIU	FP UNDERFLOW ERROR
14	W/FIUV	FP UNDEFINED-VARIABLE/(-0) ERROR
16	W/FMM	FP MAINTENANCE TRAP

NOTE - IN "FEC" CODE TYPEOUTS IN ERROR MESSAGES ONLY THE LOW ORDER BYTE IS USED - IGNORE THE PROGRAM FLAG BIT IN THE UPPER BYTE.

```
                        FLOATING POINT DATA:

                        IN FLOAT MODE (FD=0), IS 2-16.  BIT WORDS, 32.  BITS
                        IN DOUBLE MODE (FD=1), IS 4-16.  BIT WORDS, 64.  BITS

                        FIRST WORD:  (BOTH F, D MODES)
                                B15=SIGN OF NUMBER (1/-, 0/+)
                                B14:07=EXPONENT, 8.BITS, FROM -128./+127.
                                B06:00=FRACTION, 7.BITS
                        SECOND WORD:  (BOTH F, D MODES)
                                B15:00=FRACTION, 16.BITS
                        THIRD, FOURTH WORDS:  (ONLY D MODE)
                                B15:00, B15:00=FRACTION, 32. BITS

                        IN F MODE, THE COMPOSITE 24. BIT FRACTION
                        IS FORMED BY:

                                .1#[WORD1-BIT<06:00>]#[WORD2-BIT<15:00>]

                        IN D MODE, THE COMPOSITE 56. BIT FRACTION
                        IS FORMED BY:

                                .1#[WORD1-BIT<06:00>]#[WORD2-BIT<15:00>]
                                        #[WORD3-BIT<15:00>]#[WORD4-BIT<15:00>]

                        FOR A MORE DETAILED OPERATION/EXPLANATION  OF  FLOATING  POINT
                        DATA  FORMATS  AND  OPERATIONS,  SEE  THE  PDP-11/6X PROCESSOR
                        HANDBOOK SECTION ON THE FLOATING POINT INSTRUCTION SET.


6.2                     RECOVERY

                        RECOVERY  FROM  ERRORS  HAS  BEEN  ATTEMPTED  TO  BE  MADE  AS
                        AUTOMATIC AND EFFORTLESS AS POSSIBLE.  HOWEVER, IN MANY CASES,
                        DUE TO THE NATURE OF THE ERROR, THE PROGRAM MAY  NOT  EVEN  BE
                        ABLE  TO BE RUN (EG, IF THE FLOATING POINT MODULE IS IN A HUNG
                        STATE, AND CAN NEVER ENTER THE READY STATE TO ACCEPT A NEW FPP
                        INSTRUCTION).   AT THIS POINT, SOLVING THE PROBLEM IS A DIRECT
                        FUNCTION OF THE OPERATORS INGENUITY.   THIS  TEST  SERIES  HAS
                        BEEN  DESIGNED  TO  TEST  THE FLOATING POINT PROCESSOR SO THAT
                        THESE TYPES OF FAILURES TO RUN WILL  BE  MINIMAL.   THE  TESTS
                        HAVE  BEEN PLACED IN A SPECIFICALLY STRUCTURED SEQUENCE IN THE
                        PROGRAM TO IMPLEMENT THIS STRATEGY:  TESTING  THE  MOST  BASIC
                        ELEMENTS   FIRST,   PROCEEDING   UPWARD  IN  COMPLEXITY  AFTER
                        ESTABLISHING THEIR CORRECT  OPERATION.   THIS  IS  WHY  IT  IS
                        EXTREMELY  IMPORTANT  THAT THE FLOATING POINT TEST PROGRAMS BE
                        (1) RUN IN THE PRESCRIBED ORDER, AND (2) ONLY  BE  STARTED  AT
                        THEIR  BEGINNING  ADDRESS  (USUALLY 200(8)).  THE PROGRAM WILL
                        DISPLAY, AT AN ERROR, THE MOST PERTINENT INFORMATION  RELATING
                        TO THE ERROR, AND A BRIEF EXPLANATION OF THE FAILING FUNCTION.


6.3                     CAUSES
```

THESE TEST PROGRAMS ARE NOT HARDWARE ORIENTED, AND AS SUCH IT IS NOT POSSIBLE TO CALL OUT PARTICULAR HARDWARE AREAS AND MODULES RELATING TO A GIVEN FUNCTIONAL FAILURE. HARDWARE DIAGNOSIS FOR A PARTICULAR MACHINE MUST BE DONE USING THE APPROPRIATE ENGINEERING ROM FLOWS AND PRINTS, ALONG WITH THE KNOWN FUNCTIONAL ERRORS (AS DETECTED BY THE PROGRAMS). THIS IS THE INTENT UNDER WHICH THESE INSTRUCTION TESTS WERE DESIGNED AND CODED.

7. RESTRICTIONS

7.1 STARTING

THE PROGRAM MUST BE STARTED AT LOCATION 200(8) ALWAYS.

7.2 OPERATIONAL

THERE ARE NO OPERATIONAL RESTRICTIONS.

8. MISCELLANEOUS

8.1 EXECUTION TIME

	AVERAGE EXECUTION TIME PER PASS	
MODEL	SHORTEST PASS	LONGEST PASS
PDP-11/6X	1 SEC	1 MIN:20 SEC
PDP-11/6X W/FP11-E	1 SEC	X MIN:XX SEC

SEC = SECONDS / MIN = MINUTES

SHORTEST PASS ::= NO ITERATIONS, USING SWR=(004000)

LONGEST PASS ::= 2000(10) ITERATIONS/TEST, USING SWR=(000000)

8.2 STACK POINTER

THE STACK POINTER IS SET TO 1100(8) AT THE START OF EACH PASS. IF ALL IS OPERATING CORRECTLY, IT SHOULD ALSO BE THIS VALUE AT THE START OF EACH TEST, AND AT THE END OF A PASS.

8.3 POWER FAIL

THE TESTS MAY BE POWER FAILED AT ANY TIME. SPURIOUS ERROR

MESSAGES MAY OCCUR IF THE FAILURE OCCURRED WHILE THE F.P.U. WAS EXECUTING A FUNCTION, AS NONE OF ITS REGISTERS (FPS, FEC, FEA, ACCUMULATORS) ARE SAVED IN THE EVENT OF A POWER FAILURE. HOWEVER, THESE MESSAGES SHOULD ONLY OCCUR ONCE (IF AT ALL) IMMEDIATELY AFTER POWER IS RESTORED. WHEN POWER IS RESTORED, "POWER" IS TYPED ON THE CONSOLE AND EXECUTION CONTINUES WHERE IT WAS INTERRUPTED.

NOTE THAT THE "VOLATILE" SWITCH REGISTER CONTENTS ARE SAVED AND RESTORED FROM THE STACK IN A POWER FAIL SEQUENCE; THEREFORE THE SWITCH REGISTER SETTINGS SHOULD NOT BE LOST OVER A POWER FAIL.

9. PROGRAM DESCRIPTION

9.1 ORGANIZATION

THESE PROGRAMS ARE ORGANIZED AS MUCH AS POSSIBLE IN A STRAIGHTFORWARD, LINEAR MANNER. THE MAIN BODY OF CODE IS STRUCTURED AS FOLLOWS:

(1) INITIALIZATION ROUTINE
    - SETS UP VECTORS, TYPES HEADER, ETC.
(2) MAIN BODY OF TESTS
    - INLINE TEST CODE, INLINE TEST CALLS
(3) END OF PASS ROUTINE
    - END OF PASS PROCESSING
(4) TEST SUBROUTINES
    - SUBROUTINES CONTAINING COMMON TEST CODE
(5) OVERHEAD ROUTINES
    - SERVICE SUBROUTINES (TYPEOUT,ETC.)

WHEREVER FEASIBLE, COMMON SECIONS OF CODE FOR WIDELY USED FUNCTIONS ARE CONDENSED INTO SUBROUTINES TO CONSERVE MEMORY. THIS INCLUDES NOT ONLY STANDARD SERVICE ROUTINES (SUCH AS SCOPE, ERROR, AND ASCII TYPEOUT), BUT ALSO TESTING ROUTINES WHICH PERFORM VERY SIMILAR FUNCTIONS. THUS IN MANY CASES (THE "ADDF" INSTRUCTION TESTING, FOR EXAMPLE) A SINGLE BODY OF CODE (A SUBROUTINE) IS USED TO PERFORM ALL THE FUNCTIONAL TESTS, WITH A VARIABLE PARAMETER LIST PASSED AT EACH CALL CONTAINING THE DATA OPERANDS AND EXPECTED RESULT FOR EACH INDIVIDUAL TEST. THIS CONSTRUCTION FACILITATES THE ADDITION/DELETION OF TESTS (SHOULD THAT EVER BE NECESSARY), AND ALSO GREATLY CONSERVES MEMORY SPACE REQUIREMENTS WHEN A LARGE NUMBER OF CALLS TO A GIVEN BODY OF CODE ARE REQUIRED.

THE INDIVIDUAL TESTS WITHIN EACH PROGRAM HAVE ALSO BEEN SEQUENCED IN A PARTICULAR ORDER TO FACILITATE THE DETECTION AND RESOLUTION OF ERRORS AS QUICKLY AS POSSIBLE. EACH OF THE TESTS BEGINS AS SIMPLY AS POSSIBLE, FIRST TESTING THE MOST

BASIC ELEMENTS. MORE COMPLEX ELEMENTS ARE TESTED AFTERWARDS, EMPLOYING A PHILOSOPHY THAT THE SIMPLER THE TEST, THE BETTER THE RESOLUTION. ALL FUNCTIONS ARE EVENTUALLY TESTED, BUT HOPEFULLY MOST ERRORS WILL BE CAUGHT AND CORRECTED EARLY. A MUCH MORE DETAILED ANALYSIS OF THE SEQUENCE OF TESTS PERFORMED IS PRESENTED IN SECTION 9.2.

9.2 TEST DESCRIPTION

THIS DIAGNOSTIC CONSISTS OF A NUMBER OF TESTS TO EXERCISE COMBINATIONS OF FLOATING POINT INSTRUCTIONS, USING VARIOUS ADDRESS MODES, IN A MORE OR LESS APPLICATION-PROGRAM-TYPE ENVIRONMENT.

THE FIRST SECTION OF TESTS CONTAINS COMMONLY ENCOUNTERED SEQUENCES OF FLOATING POINT CODE. VARIOUS OPERANDS, ADDRESS MODES, AND FLOATING POINT ENVIRONMENTS (I.E., F/D MODES) ARE EMPLOYED. WHEN PRESENT, THE LINE AND/OR PROGRAMMABLE CLOCKS WILL BE EMPLOYED TO GENERATE I/O INTERRUPTS TO CHECK THE FP INSTRUCTION RESTART FEATURE.

THE NEXT SECTION OF CODE CHECKS FLOATING POINT EXCEPTION CONDITIONS. ALL EXCEPTION CONDITIONS ARE GENERATED, AND THE ENABLED/DISABLED AND TRAP/NO-TRAP MODES USED TO VERIFY CORRECT OPERATION.

THE LAST SECTION OF CODE CONSISTS OF A MORE COMPLEX EXERCISER SECTION THAN THAT IN THE FIRST SECTION, TO CHECK A MORE COMPLICATED SERIES OF FLOATING POINT OPERATIONS.

9.3 SUBROUTINE ABSTRACTS

9.3.1 TRAPCATCHER

THE TRAPCATCHER IS A SERIES OF INSTRUCTIONS OCCUPYING THE INTERRUPT VECTOR AREA OF MEMORY. IT CONSISTS OF THE SEQUENCE:

```
        .WORD   .+2     ;PC AFTER TRAP
        .WORD   0       ;PS AFTER TRAP
```

PLACED AT EACH VECTOR ADDRESS IN LOCATIONS 4-776(8) OF MEMORY. THE FIRST WORD OF EACH PAIR ("PC AFTER TRAP") POINTS TO THE SECOND WORD, WHICH SERVES A DUAL PURPOSE AS
(1) THE NEW LOADED PS (ALL ZEROS), AND (2) THE NEXT INSTRUCTION TO EXECUTE (0=HALT).

WHEN THE PROGRAM IS EXECUTING, ANY REQUIRED VECTORS ARE SET UP IN THE VECTOR AREA WITH APPROPRIATE VALUES; THE OTHERS BEING LEFT IN THE "TRAPCATCHER" STATE. THUS, IF AN UNEXPECTED TRAP EVER OCCURS IN THE MACHINE, IT WILL BE CAUGHT, AND THE MACHINE SUBSEQUENTLY HALTED, DISPLAYING THE VECTOR ADDRESS * PLUS FOUR

* IN THE ADDRESS LIGHTS.

9.3.2 SCOPE ROUTINE - $SCOPE

THE SCOPE ROUTINE IS ENTERED FROM THE FIRST INSTRUCTION OF EACH TEST IN THE PROGRAM. (NOTE THAT BY DEFINITION, A "TEST" WILL BE DESIGNATED AS THE SECTION OF CODE BETWEEN TWO "SCOPE" STATEMENTS.) THIS ROUTINE PROVIDES THE OVERHEAD CODE NECESSARY TO IMPLEMENT SEVERAL OF THE SWITCH REGISTER CONTROL OPTIONS. UPON ENTRANCE TO A TEST, THE SCOPE STATEMENT AT THE BEGINNING SETS UP CERTAIN LOCATIONS (SEE BELOW) TO SPECIFY THE CURRENT TEST NUMBER AND LOOPING ADDRESS (FOR ITERATIONS). CONTROL IS THEN PASSED TO THE ACTUAL TEST CODE, PERFORMING THE DESIRED TEST. UPON EXIT, THE SCOPE STATEMENT OF THE NEXT TEST IS ENTERED, WHICH DETERMINES WHETHER TO (1) LOOP BACK TO THE PREVIOUS TEST (EG. FOR ITERATIONS) OR (2) INITIALIZE FOR THE NEXT TEST (AS DESCRIBED EARLIER, ABOVE).

ENTRANCE TO THE SCOPE ROUTINE IS VIA AN "IOT" TRAP CALL THROUGH LOCATION 20(8), (FROM THE SCOPE=IOT EQUATE). DEPENDING UPON THE SWITCH SETTINGS (SEE 5.2), CODE IS PRESENT TO: LOAD THE FP11 MICRO BREAK REGISTER, LOOP ON THE CURRENTLY EXECUTING TEST, LOOP ON A SPECIFIC TEST, PERFORM ITERATIONS OF EACH TEST, AND SET UP ADDRESSES FOR POSSIBLE LOOPING ON ERRORS. IMPORTANT VALUES USED IN THIS ROUTINE ARE:

- $MXCNT - MAXIMUM NUMBER OF ITERATIONS PER TEST (GENERALLY WILL BE 2000(10))
- $TSTNM - A COUNTER INDICATING THE NUMBER (1-377(8)) OF THE TEST CURRENTLY BEING EXECUTED
- $LPADR - CONTAINS THE ADDRESS TO WHICH THE SCOPE ROUTINE 10370 WILL LOOP, IF THE CURRENT TEST IS BEING LOOPED UPON
- $LPERR - CONTAINS THE ADDRESS TO WHICH THE ERROR ROUTINE (SEE 9.3.3) WILL LOOP, IF AN ERROR OCCURS AND THE LOOPING ON AN ERROR OPTION IS SPECIFIED IN THE SWITCHES. SET UP BY SCOPE, GENERALLY WILL BE THE SAME AS $LPADR, ABOVE.

9.3.3 ERROR ROUTINE - $ERROR

THE ERROR ROUTINE IS ENTERED WHEN THE TEST CODE HAS DETERMINED THAT AN ERROR HAS OCCURRED AS PART OF A TEST. THROUGH USE OF THIS ROUTINE, THE TEST HAS A MEANS OF SIGNALING AN ERROR TO THE 10550 OPERATOR/MONITOR; AND IMPLEMENTING THE CONTROL FUNCTIONS FOR HALTING ON ERROR, BELL ON ERROR, AND LOOPING ON ERROR. IN ADDITION, THE ERROR ROUTINE HAS THE PROVISION TO TYPE OUT ON THE OPERATOR'S CONSOLE A MESSAGE BRIEFLY EXPLAINING THE ERROR, AND SOME OF THE MOST PERTINENT DATA VALUES TO HELP DIAGNOSE THE CAUSE (SEE SECTION 6.2).

THE CALLING MECHANISM IS SIMILAR TO THAT EMPLOYED FOR THE SCOPE ROUTINE (VIA A TRAP), EXCEPT IN THIS INSTANCE, THE "EMT"

INSTRUCTION IS USED, TRAPPING THROUGH LOCATION 30(8). (NOTE THE EQUATE ERROR N=EMT N). THE LOWER BYTE OF THE EMT INSTRUCTION IS CAPABLE OF TRANSMITTING A NUMBER FROM 0-377(8), WHICH WILL BE TERMED THE "ERROR ITEM NUMBER." THIS NUMBER DETERMINES WHICH ERROR MESSAGE, AND ASSOCIATED DATA VALUES WILL BE TYPED OUT WHEN A PARTICULAR ERROR IS SIGNALED. IF THIS NUMBER IS ZERO, JUST THE PC OF THE CALLING "ERROR" INSTRUCTION WILL BE TYPED, OTHERWISE, THE NUMBER IS USED AS AN INDEX THROUGH THE ERROR TABLE ($ERRTB) TO FIND THE APPROPRIATE VALUES TO TYPE (SEE PROGRAM LISTING FOR FURTHER DETAILS).

IMPORTANT VALUES USED IN THIS ROUTINE ARE:

- EREG0 THRU EREG7 - CONTENTS OF GENERAL REGISTERS R0 THRU R7 JUST BEFORE ERROR CALL
- SERTTL - COMULATIVE NUMBER OF ERRORS ENCOUNTERED TO DATE
- SERRPC - CONTAINS THE PC OF THE "ERROR" INSTRUCTION JUST EXECUTED
- SLPERR - CONTAINS THE ADDRESS WHICH WILL BE LOOPED UPON FOR THE ERROR LOOPING FACILITY

9.3.4 ERROR MESSAGE TYPEOUT ROUTINE - $TYPERR

THIS ROUTINE ($TYPERR ENTRY POINT) IS CALLED BY THE ERROR PROCESSING ROUTINE DESCRIBED IN 9.3.3 ABOVE. ITS PURPOSE IS TO IMPLEMENT THE ERROR MESSAGE/DATA VALUE ERROR TYPEOUT FACILITY. THE SUBROUTINE WILL, GIVEN THE INDEXING BYTE FROM THE ERROR CALL INSTRUCTION, PICK UP THE CORRECT ERROR MESSAGE VECTOR FROM $ERRTB (ERROR TABLE), AND TYPE OUT THE ERROR MESSAGE, DATA HEADER, AND DATA VALUES ON THE CONSOLE.

9.3.5 TYPE ROUTINE - $TYPE

THIS ROUTINE IS THE STANDARD SYSTEM TYPEOUT ROUTINE FOR ASCII SINGLE-CHARACTER-PER-BYTE STRINGS. IT IS CALLED THROUGH A TRAP INSTRUCTION WITH THE NEXT WORD CONTAINING THE ADDRESS OF THE FIRST CHARACTER IN THE STRING. TYPING TERMINATES WHEN AN ALL-ZERO BYTE IS FOUND. HORIZONTAL TAB STOPS ARE ALSO AUTOMATICALLY PLACED.

9.3.6 OCTAL NUMBER TYPE ROUTINE - $TYPOC

THIS ROUTINE CONVERTS THE TOP NUMBER ON THE STACK TO A 6-DIGIT OCTAL REPRESENTATION, AND TYPES IT ON THE CONSOLE USING THE TYPE ROUTINE $TYPE. SEE LISTING FOR OPTIONS AND FURTHER DETAILS.

9.3.7 POWER UP AND DOWN ROUTINES - $PWRUP AND $PWRDN

THESE TWO ROUTINES ARE ENTERED FOR THE POWER UP AND DOWN CONDITIONS, RESPECTIVELY. THE POWER DOWN ROUTINE ($PWRDN) SAVES THE GENERAL REGISTERS AND STACK POINTER. THE POWER UP

ROUTINE ($PWRUP) CORRESPONDINGLY RESTORES THE REGISTERS, STACK POINTER, AND TYPES THE MESSAGE "POWER" WHEN POWER IS RESTORED. THE VOLATILE INTERNAL SWITCH REGISTER IS ALSO SAVED/RESTORED BY THIS ROUTINE.

9.3.8 END OF PASS ROUTINE - $EOP

THE END OF PASS ROUTINE COUNTS THE NUMBER OF PASSES PERFORMED, DINGS THE BELL/TYPES A MESSAGE (IF ENABLED), SETS/CLEARS THE T-BIT (IF ENABLED), AND ALSO INTERFACES TO THE MONITOR, IF PRESENT. IT ALSO OPTIONALLY LOOPS FOR A NUMBER OF SUBPASSES BEFORE SIGNALLING AN END OF PASS CONDITION.

10. ACT/APT/XXDP

10.1 ACT COMPATIBILITY

THIS PROGRAM WILL RUN UNDER THE ACT SYSTEM.

10.2 APT COMPATIBILITY

THIS PROGRAM WILL RUN UNDER THE APT SYSTEM MONITOR. ALL NECESSARY SOFTWARE COMMUNICATION HOOKS ARE PRESENT.

10.3 XXDP COMPATIBILITY

FOR XXDP MEDIA COMPATIBILITY, THE TOP 2K WORDS OF THE 16K WORD MINIMUM MEMORY AREA ARE NOT DISTURBED DURING EXECUTION.

.ENDR

```
                                .TITLE  FPU INSTR EXERCISER
                                ;*COPYRIGHT (C) 1976
                                ;*DIGITAL EQUIPMENT CORP.
                                ;*MAYNARD, MASS. 01754
                                ;*
                                ;*PROGRAM BY DONALD NORTH
                                ;*
                                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                                ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
                                ;*

                                .SBTTL  OPERATIONAL SWITCH SETTINGS
                                ;*
                                ;*      SWITCH  OCTAL           USE
                                ;*      ------  ------          ------------------
                                ;*        15    100000          HALT ON ERROR
                                ;*        14    040000          LOOP ON CURRENTLY EXECUTING TEST
                                ;*        13    020000          INHIBIT ERROR TYPEOUTS
                                ;*        12    010000          INHIBIT STATUS TYPEOUTS
                                ;*        11    004000          INHIBIT ITERATIONS
                                ;*        10    000000          0=BELL ON PASS END
                                ;*              002000          1=BELL ON ERROR
                                ;*         9    001000          LOOP ON ERROR
                                ;*         8    000400          LOOP ON TEST NUMBER IN "SLPTST"
                                ;*         1    000000          0=TEST HFP/WFP ALTERNATELY EACH PASS
                                ;*              000002          1=TEST ONLY UNIT SPECIFIED IN SW<00>
                                ;*         0    000002          0=SELECT HFP, IF SW<01>=1
                                ;*              000003          1=SELECT WFP, IF SW<01>=1

                                .SBTTL  BASIC DEFINITIONS

                                ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
001100                          STACK=  1100
                                .EQUIV  EMT,ERROR       ;;BASIC DEFINITION OF ERROR CALL
                                .EQUIV  IOT,SCOPE       ;;BASIC DEFINITION OF SCOPE CALL

                                ;*MISCELLANEOUS DEFINITIONS
000011                          HT=     11              ;;CODE FOR HORIZONTAL TAB
000012                          LF=     12              ;;CODE FOR LINE FEED
000015                          CR=     15              ;;CODE FOR CARRIAGE RETURN
000200                          CRLF=   200             ;;CODE FOR CARRIAGE RETURN-LINE FEED
177776                          PS=     177776          ;;PROCESSOR STATUS WORD
                                .EQUIV  PS,PSW
177774                          STKLMT= 177774          ;;STACK LIMIT REGISTER
177772                          PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
177570                          DSWR=   177570          ;;HARDWARE SWITCH REGISTER
177570                          DDISP=  177570          ;;HARDWARE DISPLAY REGISTER

                                ;*GENERAL PURPOSE REGISTER DEFINITIONS
000000                          R0=     %0              ;;GENERAL REGISTER
000001                          R1=     %1              ;;GENERAL REGISTER
000002                          R2=     %2              ;;GENERAL REGISTER
000003                          R3=     %3              ;;GENERAL REGISTER
000004                          R4=     %4              ;;GENERAL REGISTER
000005                          R5=     %5              ;;GENERAL REGISTER
```

```
                        R6=     %6              ;;GENERAL REGISTER
                        R7=     %7              ;;GENERAL REGISTER
                        SP=     %6              ;;STACK POINTER
                        PC=     %7              ;;PROGRAM COUNTER

                        ;*PRIORITY LEVEL DEFINITIONS
                        PR0=    0               ;;PRIORITY LEVEL 0
                        PR1=    40              ;;PRIORITY LEVEL 1
                        PR2=    100             ;;PRIORITY LEVEL 2
                        PR3=    140             ;;PRIORITY LEVEL 3
                        PR4=    200             ;;PRIORITY LEVEL 4
                        PR5=    240             ;;PRIORITY LEVEL 5
                        PR6=    300             ;;PRIORITY LEVEL 6
                        PR7=    340             ;;PRIORITY LEVEL 7

                        ;*"SWITCH REGISTER" SWITCH DEFINITIONS
                        SW15=   100000
                        SW14=   40000
                        SW13=   20000
                        SW12=   10000
                        SW11=   4000
                        SW10=   2000
                        SW09=   1000
                        SW08=   400
                        SW07=   200
                        SW06=   100
                        SW05=   40
                        SW04=   20
                        SW03=   10
                        SW02=   4
                        SW01=   2
                        SW00=   1
                        .EQUIV  SW09,SW9
                        .EQUIV  SW08,SW8
                        .EQUIV  SW07,SW7
                        .EQUIV  SW06,SW6
                        .EQUIV  SW05,SW5
                        .EQUIV  SW04,SW4
                        .EQUIV  SW03,SW3
                        .EQUIV  SW02,SW2
                        .EQUIV  SW01,SW1
                        .EQUIV  SW00,SW0

                        ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
                BIT15=  100000
                BIT14=  40000
                BIT13=  20000
                BIT12=  10000
                BIT11=  4000
                BIT10=  2000
                BIT09=  1000
                BIT08=  400
                BIT07=  200
                BIT06=  100
                BIT05=  40
                BIT04=  20
```

```
                        
000010                  BIT03=  10
000004                  BIT02=  4
000002                  BIT01=  2
000001                  BIT00=  1
                        .EQUIV  BIT09,BIT9
                        .EQUIV  BIT08,BIT8
                        .EQUIV  BIT07,BIT7
                        .EQUIV  BIT06,BIT6
                        .EQUIV  BIT05,BIT5
                        .EQUIV  BIT04,BIT4
                        .EQUIV  BIT03,BIT3
                        .EQUIV  BIT02,BIT2
                        .EQUIV  BIT01,BIT1
                        .EQUIV  BIT00,BIT0
                        
                        ;*BASIC "CPU" TRAP VECTOR ADDRESSES
000004                  ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
000014                  TBITVEC=14              ;;"T" BIT
000014                  TRTVEC= 14              ;;TRACE TRAP
000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
000020                  IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
000024                  PWRVEC= 24              ;;POWER FAIL
000030                  EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
000034                  TRAPVEC=34              ;;"TRAP" TRAP
000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
                        
                        ;*MED INSTR EQUATES
076600                  MED=    076600          ;OPCODE
                        
000022                  RWHAMI= 022             ;READ WHAMI
                        
000144                  RFLAG=  144             ;READ FLAGS
000344                  WFLAG=  344             ;WRITE FLAGS
                        
                        ;*LINE CLOCK VECTORS, ADDRESSES
177546                  LKS=    177546          ;STATUS                 KW11-L LINE CLOCK
000100                  LKV=    100             ;INTERRUPT VECTOR
                        
172540                  PLKS=   172540          ;STATUS                 KW11-P PROG LINE CLOCK
172542                  PLKR=   172542          ;COUNT SET REGISTER
172544                  PLKD=   172544          ;READ COUNT REGISTER
000104                  PLKV=   104             ;INTERRUPT VECTOR
                        
                        ;*FLOATING POINT INTERRUPT VECTOR
000244                  FPPVEC= 244
                        
                        ;*FLOATING POINT REGISTER DEFINITIONS
000000                  AC0=    %0
000001                  AC1=    %1
000002                  AC2=    %2
000003                  AC3=    %3
000004                  AC4=    %4
000005                  AC5=    %5
```

```
                                        .SBTTL  TRAP CATCHER
                                
                000000                          .=0
                                        ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
                                        ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                        ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
                000174                  .=174
000174  000000                  DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
000176  000000                  SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
                                        .SBTTL  STARTING ADDRESS(ES)
000200  000137  002100                  JMP     @#START ;;JUMP TO STARTING ADDRESS OF PROGRAM
                                
                                        .SBTTL  ACT11 HOOKS
                                
                                        ;;**************************************************************
                                        ;HOOKS REQUIRED BY ACT11
                000204                  $SVPC=.                 ;SAVE PC
                000046                  .=46
000046  013730                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
                000052                  .=52
000052  000000                          .WORD   0               ;;2)SET LOC.52 TO ZERO
                000204                  .=$SVPC                 ;; RESTORE PC
                001000                  .=1000
                                        .SBTTL  APT PARAMETER BLOCK
                                
                                        ;;**************************************************************
                                        ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                                        ;;**************************************************************
                001000                          .$X=.   ;;SAVE CURRENT LOCATION
                000024                  .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
000024  000200                  200     ;;FOR APT START UP
                000044                  .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
000044  001000                  $APTHDR ;;POINT TO APT HEADER BLOCK
                001000                  .=.$X   ;;RESET LOCATION COUNTER
                                        ;;**************************************************************
                                        ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
                                        ;INTERFACE SPEC.
                                
001000                          $APTHD:
001000  000000                  $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
001002  001244                  $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
001004  000001                  $TSTM:  .WORD   1       ;;RUN TIM OF LONGEST TEST
001006  000001                  $PASTM: .WORD   1       ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
001010  000000                  $UNITM: .WORD   0       ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
001012  000014                          .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
```

```
                                .SBTTL  COMMON TAGS

                                ;;**********************************************************************
                                ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
                                ;*USED IN THE PROGRAM.

                        001100          .=1100
001100                  $CMTAG:                         ;;START OF COMMON TAGS
001100  000000                  .WORD   0
001102  000000          $TSTNM: .WORD   0               ;;CONTAINS THE TEST NUMBER
001104  000000          $ERFLG: .WORD   0               ;;CONTAINS ERROR FLAG
001106  000000          $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
001110  000000          $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
001112  000000          $LPTST: .WORD   0               ;;CONTAINS TEST NUMBER TO LOOP UPON
001114  000000          $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
001116  000000          $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
001120  000000          $ITEMB: .WORD   0               ;;CONTAINS ITEM CONTROL BYTE
001122  000001          $ERMAX: .WORD   1               ;;CONTAINS MAX. ERRORS PER TEST
001124  000000          $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001126  000000          $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
001130  000000          $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
001132  000000          $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
001134  000000          $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
001136  000000                  .WORD   0               ;;RESERVED--NOT TO BE USED
001140  000000                  .WORD   0
001142  000             $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
001143  000             $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
001144  000000                  .WORD   0
001146  177570          SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
001150  177570          DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
001152  177560          $TKS:   177560                  ;;TTY KBD STATUS
001154  177562          $TKB:   177562                  ;;TTY KBD BUFFER
001156  177564          $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
001160  177566          $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
001162  000             $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
001163  002             $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001164  012             $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001165  000             $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001166  000000          $REGAD: .WORD   0               ;;CONTAINS THE ADDRESS FROM

                                                        ;;WHICH  ($REG0) WAS OBTAINED
001170  000000          $REG0:  .WORD   0               ;;CONTAINS (($REGAD)+0)
001172  000000          $REG1:  .WORD   0               ;;CONTAINS (($REGAD)+2)
001174  000000          $REG2:  .WORD   0               ;;CONTAINS (($REGAD)+4)
001176  000000          $REG3:  .WORD   0               ;;CONTAINS (($REGAD)+6)
001200  000000          $REG4:  .WORD   0               ;;CONTAINS (($REGAD)+10)
001202  000000          $REG5:  .WORD   0               ;;CONTAINS (($REGAD)+12)
001204  000000          $REG6:  .WORD   0               ;;CONTAINS (($REGAD)+14)
001206  000000          $REG7:  .WORD   0               ;;CONTAINS (($REGAD)+16)
001210  000000          $TMP0:  .WORD   0               ;;USER DEFINED
001212  000000          $TMP1:  .WORD   0               ;;USER DEFINED
001214  000000          $TMP2:  .WORD   0               ;;USER DEFINED
001216  000000          $TMP3:  .WORD   0               ;;USER DEFINED
001220  000000          $TMP4:  .WORD   0               ;;USER DEFINED
001222  000000          $TMP5:  .WORD   0               ;;USER DEFINED
001224  000000          $TMP6:  .WORD   0               ;;USER DEFINED
001226  000000          $TMP7:  .WORD   0               ;;USER DEFINED
```

```
1017  001230  000000                  $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
1018  001232  000000                  $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
1019  001234  177607  000377          $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
1020  001240     077                  $QUES:  .ASCII  /?/             ;;QUESTION MARK
1021  001241     015                  $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
1022  001242  000012                  $LF:    .ASCIZ  <12>            ;;LINE FEED
1023                                  ;;*****************************************************************
1024                                  .SBTTL  APT MAILBOX-ETABLE
1025
1026                                  ;;*****************************************************************
1027                                  .EVEN
1028  001244                          $MAIL:                          ;;APT MAILBOX
1029  001244  000000                  $MSGTY: .WORD   AMSGTY           ;;MESSAGE TYPE CODE
1030  001246  000000                  $FATAL: .WORD   AFATAL           ;;FATAL ERROR NUMBER
1031  001250  000000                  $TESTN: .WORD   ATESTN           ;;TEST NUMBER
1032  001252  000000                  $PASS:  .WORD   APASS            ;;PASS COUNT
1033  001254  000000                  $DEVCT: .WORD   ADEVCT           ;;DEVICE COUNT
1034  001256  000000                  $UNIT:  .WORD   AUNIT            ;;I/O UNIT NUMBER
1035  001260  000000                  $MSGAD: .WORD   AMSGAD           ;;MESSAGE ADDRESS
1036  001262  000000                  $MSGLG: .WORD   AMSGLG           ;;MESSAGE LENGTH
1037  001264                          $ETABLE:                        ;;APT ENVIRONMENT TABLE
1038  001264     000                  $ENV:   .BYTE   AENV            ;;ENVIRONMENT BYTE
1039  001265     000                  $ENVM:  .BYTE   AENVM           ;;ENVIRONMENT MODE BITS
1040  001266  000000                  $SWREG: .WORD   ASWREG          ;;APT SWITCH REGISTER
1041  001270  000000                  $USWR:  .WORD   AUSWR           ;;USER SWITCHES
1042  001272  000000                  $CPUOP: .WORD   ACPUOP          ;;CPU TYPE,OPTIONS
1043                                  ;*                              BITS 15-11=CPU TYPE
1044                                  ;*                                      11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
1045                                  ;*                                      11/70=06,PDQ=07,Q=10
1046                                  ;*                              BIT 10=REAL TIME CLOCK
1047                                  ;*                              BIT  9=FLOATING POINT PROCESSOR
1048                                  ;*                              BIT  8=MEMORY MANAGEMENT
1049  001274                          $ETEND:
1050                                  .MEXIT
```

```
                                .SBTTL  ERROR POINTER TABLE

001274                          $ERRTB:

                                ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
                                ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
                                ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
                                ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
                                ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

                                ;*      EM              ;;POINTS TO THE ERROR MESSAGE
                                ;*      DH              ;;POINTS TO THE DATA HEADER
                                ;*      DT              ;;POINTS TO THE DATA
                                ;*      DF              ;;POINTS TO THE DATA FORMAT
                                        ;*NOTE:  ERROR VECTOR TABLE ($ERRTB) HAS BEEN MODIFIED,
                                        ;*       ELIMINATING UNUSED VALUE FOR DATA FORMAT POINTER.
                                        ;*       ERROR TYPING ROUTINE HAS ALSO BEEN MODIFIED
                                        ;*       ACCORDINGLY.
001274  017071  020034  020366  EMV001: .WORD   EMA,DHA,DTA      ;FPS BAD
001302  017115  020041  020320  EMV002: .WORD   EMB,DHB,DTB      ;FEC BAD
001310  017141  020046  020400  EMV003: .WORD   EMC,DHC,DTC      ;FEA BAD
001316  017165  020053  020324  EMV004: .WORD   EMD,DHD,DTD      ;PS CC-S BAD
001324  017217  020065  020326  EMV005: .WORD   EME,DHE,DTE      ;BAD ADDR IN R0
001332  017241  020072  020332  EMV006: .WORD   EMF,DHF,DTF      ;BAD SP
001340  017441  000000  000000  EMV007: .WORD   EMP,0,0          ;DIDNT TRAP, SHOULD HAVE
001346  017263  020126  020352  EMV010: .WORD   EMH,DHI,DTI      ;RESULT BAD - D
001354  017263  020112  020344  EMV011: .WORD   EMH,DHH,DTH      ;RESULT BAD - F, L
001362  017263  020104  020340  EMV012: .WORD   EMH,DHG,DTG      ;RESULT BAD - I
001370  017263  020065  020326  EMV013: .WORD   EMH,DHE,DTE      ;RESULT BAD - R0
                                ;*****  VECTORS FOR TRAP/FEC TESTER ROUTINE  *****
001376  017441  000000  000000  EMV014: .WORD   EMP,0,0          ;NO TRAP, SHOULD HAVE    TRAP/FEC TSTR
001404  017527  000000  000000  EMV015: .WORD   EMQ,0,0          ;TRAPPED, SHOULDN'T HAVE
001412  017614  020236  020422  EMV016: .WORD   EMR,DHL,DTAG     ;SP OK
001420  017663  020236  020430  EMV017: .WORD   EMS,DHL,DTAH     ;STORED PC
001426  017726  020236  020436  EMV020: .WORD   EMT,DHL,DTAI     ;STORED PS
001434  017771  020236  020444  EMV021: .WORD   EMU,DHL,DTAJ     ;LOADED PS
001442  017335  020176  020372  EMV022: .WORD   EMI,DHK,DTK      ;FEC/FEA BAD
001450  017071  020156  020364  EMV023: .WORD   EMA,DHJ,DTJ      ;FPS BAD
                                ;*****  VECTOR FOR UNEXPECTED FPP TRAP  *****
001456  017365  020252  020404  EMV024: .WORD   EMJ,DHM,DTL      ;UNEXPECTED TRAP
```

```
1091
1092
1093
1094                                    .SBTTL  PROGRAM DEFINED COMMON TAGS
1095                                    ;*VARIABLES
1096  001464  000000                    FPS:     .WORD   0              ;FPS STORED HERE AFTER STFPS
1097  001466  000000                    FEC:     .WORD   0              ;FEC STORED HERE AFTER STST
1098  001470  000000                    FEA:     .WORD   0              ;FEA STORED HERE AFTER STST
1099  001472  000000                    FPPOPC:  .WORD   0              ;OLD PC SAVED HERE AFTER TRAP
1100  001474  000000                    FPPOPS:  .WORD   0              ;OLD PS SAVED HERE AFTER TRAP
1101  001476  000000                    FPPOSP:  .WORD   0              ;SP AFTER TRAP
1102  001500  000000                    EXPFEA:  .WORD   0              ;EXPECTED FEA
1103  001502  000000                    EXPRTS:  .WORD   0              ;EXPECTED RETURN ADDR
1104  001504  000000                    EXPSP:   .WORD   0              ;EXPECTED SP VALUE AFTER TRAP
1105  001506  000000                    OPCRCV:  .WORD   0              ;PC PUSHED ON STACK AFTER TRAP
1106  001510  000000                    OPSRCV:  .WORD   0              ;PS PUSHED ON STACK AFTER TRAP
1107  001512  000000                    NPSLOD:  .WORD   0              ;PS THAT WAS LOADED AFTER TRAP
1108  001514  000000                    OSPRCV:  .WORD   0              ;SP AFTER FPP TRAP
1109
1110  001516  000000                    CLKPRS:  .WORD   000000         ;BIT07=1 -> KW11-L PRESENT
1111                                                                    ;BIT15=1 -> KW11-P PRESENT
1112  001520  000000                    KW11LC:  .WORD   0              ;KW11-L COUNT OF # OF INTERRUPTS
1113  001522  000000                    KW11PC:  .WORD   0              ;KW11-P COUNT OF # OF INTERRUPTS
1114  001524  000000                    KW11PR:  .WORD   0              ;KW11-P COUNT SET
1115
1116                                    ;*REGISTER CONTENTS, AT ERROR, STORED HERE
1117  001526  000000                    EREG0:   .WORD   0
1118  001530  000000                    EREG1:   .WORD   0
1119  001532  000000                    EREG2:   .WORD   0
1120  001534  000000                    EREG3:   .WORD   0
1121  001536  000000                    EREG4:   .WORD   0
1122  001540  000000                    EREG5:   .WORD   0
1123  001542  000000                    EREG6:   .WORD   0
1124  001544  000000                    EREG7:   .WORD   0
1125
1126                                    ;*CONSTANTS
1127  001546  040200  000000  000000    FLTONE:  .WORD   040200,0,0,0
1128  001554  000000
1129  001556  177777                    D1010:   .WORD   -1
1130  001560  000000  177777  000000    D0101:   .WORD   0,-1,0
1131  001566  177777  000000  000000    D1001:   .WORD   -1,0,0
1132  001574                            DSMALL:
1133  001574  177777  000001  000000    WEIRD:   .WORD   -1,1,0,-1
1134  001602  177777
1135  001604  001560                    AD0101:  .WORD   D0101
1136  001606  001566                    AD1001:  .WORD   D1001
1137  001610  001726                    AD1000:  .WORD   D1000
1138  001612  077777  000000  177777    DBIG:    .WORD   77777,0,-1,0
1139  001620  000000
1140  001622  100000  000000  000000    DMZERO:  .WORD   100000,0,0,0
1141  001630  000000
1142  001632  001170                    A$REG0:  .WORD   $REG0
1143  001634  040252  125252  125252    DALTA:   .WORD   40252,125252,125252,125252
1144  001642  125252
1145  001644  040325  052525  052525    DALTB:   .WORD   40325,52525,52525,52525
1146  001652  052525
```

```
1147  001654  040325  052525  052525  DALTC:  .WORD   40325,52525,52525,52526
1148  001662  052526
1149  001664  040000  000000  000000  D40:    .WORD   40000,0,0,0
1150  001672  000000
1151  001674  037400  000000  000000  D37:    .WORD   37400,0,0,0
1152  001702  000000
1153  001704  040600  000000  000000  D46:    .WORD   40600,0,0,0
1154  001712  000000
1155  001714  020000  000000  000000  D20:    .WORD   20000,0,0,0
1156  001722  000000
1157  001724  000000                  D0100:  .WORD   0
1158  001726  177777  000000  000000  D1000:  .WORD   -1,0,0,0
1159  001734  000000
1160  001736  000100  000000  000000  D0100X: .WORD   100,0,0
1161  001744  000000  177777  177777  D0111:  .WORD   0,-1,-1,-1
1162  001752  177777
1163  001754  000000  054321          D5T01:  .WORD   0,54321
1164  001760  001754                  AD5T01: .WORD   D5T01
1165  001762  043661  121000  000000  F5T01:  .WORD   43661,121000,0,0
1166  001770  000000
1167
1168
1169                                  ;*MESSAGES FOR BEGIN PROGRAM/START OF PASS
1170  001772  005015  005012  042115  BGNMES: .ASCII  <15><12><12><12>"MD-11-DQFPC-A..."
1171  002000  030455  026461  050504
1172  002006  050106  026503  027101
1173  002014  027056
1174  002016  042120  026520  030461          .ASCIZ  "PDP-11/6X F.P.U. INSTRUCTION EXERCISER"<15><12>
1175  002024  033057  020130  027106
1176  002032  027120  027125  044440
1177  002040  051516  051124  041525
1178  002046  044524  047117  042440
1179  002054  042530  041522  051511
1180  002062  051105  005015     000
1181  002067     015  050012  051501  NWPAS1: .ASCIZ  <15><12>"PASS #"
1182  002074  020123  000043
```

```
                                        .SBTTL  START OF PASS ROUTINE

                                                .EVEN                   ;START ON AN EVEN BOUNDARY

                                        ;;**************************************************************************
                                                .ENABL  AMA             ;ASSEMBLE ALL RELATIVE REFERENCES AS ABSOLUTE
                                        ;;**************************************************************************

002100                                  START:
                                        .SBTTL  INITIALIZE THE COMMON TAGS
                                        ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
002100  012706  001100                          MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
002104  005026                                  CLR     (R6)+           ;;CLEAR MEMORY LOCATION
002106  022706  001146                          CMP     #SWR,R6 ;;DONE?
002112  001374                                  BNE     .-6             ;;LOOP BACK IF NO
002114  012706  001100                          MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                        ;;INITIALIZE A FEW VECTORS
002120  012737  014704  000020                  MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
002126  012737  000340  000022                  MOV     #340,@#IOTVEC+2 ;;LEVEL 7
002134  012737  015162  000030                  MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
002142  012737  000340  000032                  MOV     #340,@#EMTVEC+2 ;;LEVEL 7
002150  012737  016616  000034                  MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
002156  012737  000340  000036                  MOV     #340,@#TRAPVEC+2;LEVEL 7
002164  012737  016664  000024                  MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
002172  012737  000340  000026                  MOV     #340,@#PWRVEC+2 ;;LEVEL 7
002200  013737  013700  013672                  MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
002206  005037  001230                          CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
002212  005037  001232                          CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
002216  012737  000001  001122                  MOV     #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
002224  012737  002224  001110                  MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
002232  012737  002232  001114                  MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
                                        ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                        ;;EQUAL TO A "-1"  SETUP FOR A SOFTWARE SWITCH REGISTER.
002240  013746  000004                          MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
002244  012737  002300  000004                  MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
002252  012737  177570  001146                  MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
002260  012737  177570  001150                  MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
002266  022777  177777  176652                  CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
002274  001012                                  BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                        ;;AND  THE HARDWARE SWR IS NOT = -1
002276  000403                                  BR      65$             ;;BRANCH IF NO TIMEOUT
002300  012716  002306          64$:            MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
002304  000002                                  RTI
002306  012737  000176  001146  65$:            MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
002314  012737  000174  001150                  MOV     #DISPREG,DISPLAY
002322  012637  000004          66$:            MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

002326  005037  001252                          CLR     $PASS           ;;CLEAR PASS COUNT
002332  132737  000200  001265                  BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
002340  001403                                  BEQ     67$             ;;YES,USE NON-APT SWITCH
002342  012737  001266  001146                  MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
002350                          67$:

                                        ;SET UP FPP UNEXPECTED TRAP CATCHER - - - - - - - - -
002350  012737  014620  000244          MOV     #FPPILT,@#FPPVEC        ;NEW PC AT FPP TRAP
002356  012737  000300  000246          MOV     #PR6,@#FPPVEC+2         ;NEW PS AT FPP TRAP
```

```
1239
1240                                                    ;FIND, INITIALIZE ANY LINE CLOCKS PRESENT - - - - - - - - - -
1241  002364  005037  001516                            CLR     CLKPRS          ;CLEAR CLOCK PRESENT FLAG
1242  002370  013746  000004                            MOV     @#ERRVEC,-(SP)          ;SAVE OLD TIMEOUT VECTOR
1243  002374  013746  000006                            MOV     @#ERRVEC+2,-(SP)        ;
1244  002400  012737  002450  000004                     MOV     #NKW11L,@#ERRVEC        ;SET  NEW TIMEOUT VECTOR
1245  002406  012737  000340  000006                     MOV     #PR7,@#ERRVEC+2 ;
1246
1247  002414  005037  177546                            CLR     @#LKS           ;KW11-L THERE ? (CLEAR STATUS IF YES)
1248  002420  052737  000200  001516                     BIS     #BIT7,CLKPRS    ;YES, IT WAS
1249  002426  005037  001520                            CLR     KW11LC          ;CLEAR COUNT
1250  002432  012737  014462  000100                     MOV     #IKW11L,@#LKV   ;KW11-L SERVICE PC, PS
1251  002440  012737  000300  000102                     MOV     #PR6,@#LKV+2    ;
1252  002446  000402                                    BR      TKW11P          ;
1253  002450  062706  000004            NKW11L: ADD     #4,SP           ;NO KW11-L (TIMEOUT); POP JUNK OFF STK
1254
1255  002454  012737  002532  000004     TKW11P: MOV     #NKW11P,@#ERRVEC        ;SET NEW TIMEOUT VECTOR (SAME PS)
1256
1257  002462  005037  172540                            CLR     @#PLKS          ;KW11-P THERE ? (CLEAR STATUS IF YES)
1258  002466  052737  100000  001516                     BIS     #BIT15,CLKPRS   ;YES, IT WAS
1259  002474  005037  001522                            CLR     KW11PC          ;CLEAR COUNT
1260  002500  012737  001777  001524                     MOV     #1777,KW11PR    ;INITIAL COUNT REGISTER
1261  002506  012737  014510  000104                     MOV     #IKW11P,@#PLKV  ;KW11-P SERVICE PC, PS
1262  002514  012737  000300  000106                     MOV     #PR6,@#PLKV+2   ;
1263  002522  013737  001524  172542                     MOV     KW11PR,@#PLKR   ;SET COUNT REGISTER
1264  002530  000402                                    BR      CLKDON          ;
1265  002532  062706  000004            NKW11P: ADD     #4,SP           ;NO KW11-P (TIMEOUT); POP JUNK OFF STK
1266
1267  002536  012637  000006            CLKDON: MOV     (SP)+,@#ERRVEC+2        ;RESTORE OLD TIMEOUT VECTOR
1268  002542  012637  000004                    MOV     (SP)+,@#ERRVEC          ;
1269
1270  002546  104401  001772                            TYPE    ,BGNMES         ;ID MESSAGE AT START
1271
1272                                                    ;////////////////////////////////////////////////////////////////
1273                                                    ;       MESSAGE ON WHETHER OR NOT HFP UNIT IS PRESENT
1274
1275  002552  076600  000022                            MED     ,RWHAMI                 ;WHAMI INTO R0
1276  002556  032700  000020                            BIT     #BIT04,R0               ;IS THERE A HFP UNIT ?
1277  002562  001403                                    BEQ     66$                     ;NO, BR
1278  002564  104401  002600                            TYPE    ,64$                    ;INDICATE FP11-E PRESENT
1279  002570  000453                                    BR      NEWPAS                  ;GO FOR SUBPASS INIT
1280  002572  104401  002640            66$:    TYPE    ,65$                    ;INDICATE NO FP11-E
1281  002576  000450                                    BR      NEWPAS                  ;GO FOR SUBPASS INIT
1282
1283  002600  005015  020052  050106  64$:    .ASCIZ  <15><12>"* FP11-E HFP UNIT PRESENT *"<15><12>
1284  002606  030461  042455  044040
1285  002614  050106  052440  044516
1286  002622  020124  051120  051505
1287  002630  047105  020124  006452
1288  002636  000012
1289  002640  005015  020052  047516  65$:    .ASCIZ  <15><12>"* NO FP11-E HFP UNIT - ALL TESTS WFP ONLY *"<15><12>
1290  002646  043040  030520  026461
1291  002654  020105  043110  020120
1292  002662  047125  052111  026440
1293  002670  040440  046114  052040
1294  002676  051505  051524  053440
```

```
                                                .EVEN
                                        ;////////////////////////////////////////////////////////////////////////////////
                                        ;;**************************************************************************
                                        ;;              ;NEW PASS ENTERS HERE
                                        ;;**************************************************************************
002720  012706  001100          NEWPAS: MOV     #STACK,SP               ;RESET STACK PTR
002724  032777  010000  176214          BIT     #BIT12,@SWR             ;INHIBIT STATUS TYPEOUTS ?
002732  001011                        BNE     SUBPAS                  ;BR IF YES
002734  104401  002067                TYPE    ,NWPAS1                 ;"PASS #"
002740  013746  001252                MOV     $PASS,-(SP)             ;PASS COUNT INTO ...
002744  005216                        INC     (SP)                    ;  1-N RANGE
002746  104403                        TYPOS                           ;TYPE OCTAL
002750     006     000                .BYTE   6,0                     ;  6 DIGITS, NO LEADING ZEROS
002752  104401  001241                TYPE    ,$CRLF                  ;END THE LINE
                                        ;;**************************************************************************
                                        ;;              ;NEW SUBPASS ENTERS HERE
                                        ;;**************************************************************************
002756  076600  000022          SUBPAS: MED     ,RWHAMI                 ;GET WHAMI INTO R0
002762  032700  000020                  BIT     #BIT04,R0               ;1=HFP PRESENT, 0=NO
002766  001430                        BEQ     20$                     ;IF NO HFP, TEST WARM ONLY
002770  076f00  000144                MED     ,RFLAG                  ;GET FLAGS INTO R0
002774  032777  000002  176144        BIT     #SW01,@SWR              ;SW01:  1=HFP OR WFP TEST ONLY
003002  001413                        BEQ     1$                      ;       0=ALTERNATE HFP/WFP PER PASS
003004  032777  000001  176134        BIT     #SW00,@SWR              ;SW00:  1=WFP ONLY
003012  001403                        BEQ     2$                      ;       0=HFP ONLY
003014  042700  010000                BIC     #BIT12,R0               ;CLEAR HFP ENABLE FLAG<5> FOR WFP
003020  000402                        BR      3$                      ;
003022  052700  010000          2$:   BIS     #BIT12,R0               ;SET HFP ENABLE FLAG<5> FOR HFP
003026  076600  000344          3$:   MED     ,WFLAG                  ;REWRITE FLAGS
003032  032700  010000          1$:   BIT     #BIT12,R0               ;TEST WHO'S ENABLED: HOT, WARM
003036  001404                        BEQ     20$                     ;SET APPROPRIATE HEADER:
003040  012737  017054  015426  19$:   MOV     #ASCHOT,HOTWRM          ;"HOT: "
003046  000403                        BR      21$                     ;
003050  012737  017062  015426  20$:   MOV     #ASCWRM,HOTWRM          ;"WARM: "
003056  005037  001102          21$:   CLR     $TSTNM                  ;ALL DONE, RESET TEST NUMBER COUNTER
                                        ;#START KW11-L LINE CLOCK IF PRESENT
003062  105737  001516                TSTB    CLKPRS          ;KW11-L PRESENT ?
003066  100003                        BPL     NKWL1           ;NO
003070  052737  000100  177546        BIS     #BIT6,@#LKS     ;YES, START IT
```

```
1351
1352                                                    ;*START KW11-P PROGRAMMABLE LINE CLOCK, IF PRESENT
1353  003076  005737  001516          NKWL1:  TST     CLKPRS          ;KW11-P PRESENT ?
1354  003102  100003                  BPL     NKWP1           ;NO
1355  003104  052737  000101  172540          BIS     #BIT6+BIT0,@#PLKS       ;YES, START IT
1356
1357  003112  005037  177776          NKWP1:  CLR     @#PSW           ;SETUP FOR INTERRUPT ENABLE, PRO
1358
1359
```

```
1360
1361                                    ;;*****************************************************************
1362                                            .DSABL  AMA             ;ASSEMBLE ALL REFERENCES AS THEY ARE PRINTED
1363                                    ;;*****************************************************************
1364
1365                                    ;;*****************************************************************
1366                                    ;*TEST 1         TEST OF WRITABILITY OF FPS
1367                                    ;;*****************************************************************
1368  003116  000004                    TST1:   SCOPE
1369  003120  170127  147757                   LDFPS   #147757         ;TEST FPS WITH ALL ONES
1370  003124  170267  176334                   STFPS   FPS             ;GET STATUS
1371  003130  022767  147757  176326           CMP     #147757,FPS     ;CHECK IT
1372  003136  001401                           BEQ     .+4
1373  003140  104001                           ERROR   1               ;FPS NOT 147777
1374
1375  003142  170127  040000                   LDFPS   #40000          ;TEST FPS WITH 40000
1376  003146  170267  176312                   STFPS   FPS             ;STORE FLOATING POINT STATUS
1377  003152  022767  040000  176304           CMP     #40000,FPS      ;CHECK FLOATING POINT STATUS
1378  003160  001401                           BEQ     .+4             ;BRANCH IF OK
1379  003162  104001                           ERROR   1               ;FPS NOT EQUAL TO 40000
1380
1381  003164  170011                           SETD                    ;TEST OF DOUBLE BIT ON A 1
1382  003166  170267  176272                   STFPS   FPS             ;STORE FLOATING POINT STATUS
1383  003172  022767  040200  176264           CMP     #40200,FPS      ;CHECK FLOATING POINT STATUS
1384  003200  001401                           BEQ     .+4             ;BRANCH IF OK
1385  003202  104001                           ERROR   1               ;FPS NOT EQUAL TO 40200
1386
1387  003204  170001                           SETF                    ;TEST OF DOUBLE BIT ON A 0
1388  003206  170267  176252                   STFPS   FPS             ;STORE FLOATING POINT STATUS
1389  003212  022767  040000  176244           CMP     #40000,FPS      ;CHECK FLOATING POINT STATUS
1390  003220  001401                           BEQ     .+4             ;BRANCH IF OK
1391  003222  104001                           ERROR   1               ;FPS NOT EQUAL TO 40000
1392
1393  003224  170012                           SETL                    ;TEST OF LONG BIT ON A 1
1394  003226  170267  176232                   STFPS   FPS             ;STORE FLOATING POINT STATUS
1395  003232  022767  040100  176224           CMP     #40100,FPS      ;CHECK FLOATING POINT STATUS
1396  003240  001401                           BEQ     .+4             ;BRANCH IF OK
1397  003242  104001                           ERROR   1               ;FPS NOT EQUAL TO 40100
1398
1399  003244  170002                           SETI                    ;TEST OF LONG BIT ON A 0
1400  003246  170267  176212                   STFPS   FPS             ;STORE FLOATING POINT STATUS
1401  003252  022767  040000  176204           CMP     #40000,FPS      ;CHECK FLOATING POINT STATUS
1402  003260  001401                           BEQ     .+4             ;BRANCH IF OK
1403  003262  104001                           ERROR   1               ;FPS NOT EQUAL TO 40000
```

```
1404                                    ;;*****************************************************************
1405                                    ;*TEST 2         TEST OF CFCC
1406                                    ;;*****************************************************************
1407   003264  000004                   TST2:   SCOPE
1408   003266  170127  040017                   LDFPS   #40017          ;LOAD ALL STATUS BITS TO 1'S
1409   003272  170000                           CFCC                    ;GET THEM INTO PS
1410   003274  013700  177776                   MOV     @#PS,R0         ;GET THEM FOR TYPING
1411   003300  042700  177760                   BIC     #177760,R0      ;CLEAR JUNK
1412   003304  022700  000017                   CMP     #17,R0          ;ALL SET?
1413   003310  001401                           BEQ     .+4
1414   003312  104004                           ERROR   4               ;PS NOT 17
1415
1416   003314  170127  040012                   LDFPS   #40012          ;LOAD FPS WITH 12
1417   003320  170000                           CFCC                    ;GET INTO PS
1418   003322  013700  177776                   MOV     @#PS,R0         ;GET FOR TYPING
1419   003326  042700  177760                   BIC     #177760,R0      ;CLEAR JUNK
1420   003332  022700  000012                   CMP     #12,R0          ;SAME AS LD?
1421   003336  001401                           BEQ     .+4
1422   003340  104004                           ERROR   4               ;PS NOT 12
1423
1424   003342  170127  040005                   LDFPS   #40005          ;LOAD FPS WITH 5
1425   003346  170000                           CFCC                    ;GET BITS
1426   003350  013700  177776                   MOV     @#PS,R0         ;GET FOR TYPING
1427   003354  042700  177760                   BIC     #177760,R0      ;CLEAR JUNK
1428   003360  022700  000005                   CMP     #5,R0           ;SAME?
1429   003364  001401                           BEQ     .+4
1430   003366  104004                           ERROR   4               ;PS NOT 5
```

```
1431                                    ;;*****************************************************************
1432                                    ;*TEST 3        TEST OF LDD, STD, CMPD OF -1,0,-1,0
1433                                    ;;*****************************************************************
1434  003370  000004                    TST3:   SCOPE
1435  003372  170127  047600                    LDFPS   #47600
1436  003376  012700  001556                    MOV     #D1010,R0       ;GET ADDRESS OF DATA WORD
1437  003402  172420                            LDD     (R0)+,AC0       ;LOAD INTO AC0
1438  003404  022700  001566                    CMP     #D1010+10,R0    ;IS THE NEW ADDRESS RIGHT?
1439  003410  001401                            BEQ     .+4
1440  003412  104005                            ERROR   5               ;R0 NOT D1010+10
1441
1442  003414  010601                            MOV     SP,R1
1443  003416  162701  000010                    SUB     #10,R1
1444  003422  174046                            STD     AC0,-(SP)       ;GET THE DATA BACK
1445  003424  010600                            MOV     SP,R0           ;SAVE THE SP FOR TYPING
1446  003426  020106                            CMP     R1,SP           ;SP DECREMENTED PROPERLY?
1447  003430  001401                            BEQ     .+4
1448  003432  104006                            ERROR   6               ;SP NOT SP-10
1449
1450  003434  170267  176024                    STFPS   FPS             ;STORE FLOATING POINT STATUS
1451  003440  022767  047610  176016            CMP     #47610,FPS      ;CHECK FLOATING POINT STATUS
1452  003446  001401                            BEQ     .+4             ;BRANCH IF OK
1453  003450  104001                            ERROR   1               ;FPS NOT EQUAL TO 47610
1454
1455  003452  010602                            MOV     SP,R2           ;MOVE ANSWER TO DISPLAY
1456  003454  012703  001170                    MOV     #$REG0,R3       ;
1457  003460  012223                            MOV     (R2)+,(R3)+     ;
1458  003462  012223                            MOV     (R2)+,(R3)+     ;
1459  003464  012223                            MOV     (R2)+,(R3)+     ;
1460  003466  012223                            MOV     (R2)+,(R3)+     ;
1461  003470  021667  176062                    CMP     (SP),D1010      ;CHECK FIRST PIECE OF DATA
1462  003474  001401                            BEQ     .+4
1463  003476  104010                            ERROR   10              ;DATA IN (SP) NOT D1010
1464  003500  026667  000002  176052            CMP     2(SP),D1010+2   ;CHECK SECOND
1465  003506  001401                            BEQ     .+4
1466  003510  104010                            ERROR   10              ;DATA IN 2(SP) NOT D1010+2
1467  003512  026667  000004  176042            CMP     4(SP),D1010+4   ;CHECK THIRD
1468  003520  001401                            BEQ     .+4
1469  003522  104010                            ERROR   10              ;DATA IN 4(SP) NOT D1010+4
1470  003524  026667  000006  176032            CMP     6(SP),D1010+6   ;CHECK FOURTH
1471  003532  001401                            BEQ     .+4
1472  003534  104010                            ERROR   10              ;DATA IN 6(SP) NOT D1010+6
1473
1474  003536  062701  000010                    ADD     #10,R1
1475  003542  173426                            CMPD    (SP)+,AC0       ;RECHECK DATA AND SP
1476  003544  010600                            MOV     SP,R0           ;SAVE SP FOR TYPING
1477  003546  020106                            CMP     R1,SP           ;CHECK ADDRESS IN SP
1478  003550  001401                            BEQ     .+4
1479  003552  104006                            ERROR   6               ;SP NOT RESTORED
1480  003554  170267  175704                    STFPS   FPS             ;GET STATUS
1481  003560  170000                            CFCC                    ;NOW GET THE FP CONDITION CODES
1482  003562  001401                            BEQ     .+4             ;IF IT HALTS HERE IT MUST BE THE
1483  003564  104010                            ERROR   10              ;CMPD BECAUSE CFCC IS ALREADY CONFIRMED
```

```
1484                                    ;;*****************************************************************
1485                                    ;*TEST 4        TEST OF LDD, STD, CMPD OF 0,-1,0,-1
1486                                    ;;*****************************************************************
1487  003566  000004                    TST4:   SCOPE
1488  003570  170127  047600                    LDFPS   #47600
1489  003574  172437  001560                    LDD     @#D0101,AC0     ;LOAD 0,-1,0,-1 INTO AC0 *PIC*
1490  003600  170267  175660                    STFPS   FPS             ;STORE FLOATING POINT STATUS
1491  003604  022767  047604  175652            CMP     #47604,FPS      ;CHECK FLOATING POINT STATUS
1492  003612  001401                            BEQ     .+4             ;BRANCH IF OK
1493  003614  104001                            ERROR   1               ;FPS NOT EQUAL TO 47604
1494
1495  003616  012700  001170                    MOV     #$REG0,R0       ;ADDRESS TO BE STORED INTO
1496  003622  174010                            STD     AC0,(R0)        ;STORE IT INTO $REG0-3 *PIC*
1497  003624  026767  175730  175336            CMP     D0101,$REG0     ;FIRST WORD OK?
1498  003632  001401                            BEQ     .+4
1499  003634  104010                            ERROR   10              ;$REG0 NOT D0101
1500  003636  026767  175720  175326            CMP     D0101+2,$REG1   ;SECOND
1501  003644  001401                            BEQ     .+4
1502  003646  104010                            ERROR   10              ;$REG1 NOT D0101+2
1503  003650  026767  175710  175316            CMP     D0101+4,$REG2   ;THIRD
1504  003656  001401                            BEQ     .+4
1505  003660  104010                            ERROR   10              ;$REG2 NOT D0101+4
1506  003662  026767  175700  175306            CMP     D0101+6,$REG3   ;FOURTH
1507  003670  001401                            BEQ     .+4
1508  003672  104010                            ERROR   10              ;$REG3 NOT D0101+6
1509
1510  003674  012704  001602                    MOV     #AD0101-2,R4    ;ADDRESS-2 OF DATA
1511  003700  173474  000002                    CMPD    @2(4),AC0       ;CHECK DATA IN AC0 *PIC*
1512                                            ;NOTE:  CMPD SEES A ZERO EXP IN AC0, AND THUS
1513                                            ;  AFTER ITS EXECUTION AC0 <- (0,0,0,0),
1514                                            ;  A TRUE FLTPT ZERO
1515  003704  170267  175554                    STFPS   FPS             ;GET STATUS
1516  003710  170000                            CFCC                    ;GET CONDITION CODES
1517  003712  001401                            BEQ     .+4
1518  003714  104010                            ERROR   10              ;CMPD FAILED
```

```
1519                                    ;;*****************************************************************
1520                                    ;*TEST 5        TEST OF LDF, STF, CMPF OF -1,0
1521                                    ;;*****************************************************************
1522  003716  000004                    TST5:   SCOPE
1523                                                                ;NOTE:  AC0 = (0,0,0,0) FROM PREV TEST
1524  003720  170127  047400                   LDFPS   #47400       ;SET FLOATING MODE
1525  003724  172467  175626                   LDF     D1010,AC0    ;LOAD -1,0 INTO AC0
1526  003730  012700  001170                   MOV     #$REG0,R0    ;POINTED TO ANSWER AREA *PIC*
1527  003734  174020                           STF     AC0,(R0)+    ;STORE RESULT
1528  003736  022700  001174                   CMP     #$REG2,R0    ;INCREMENTED PROPERLY
1529  003742  001401                           BEQ     .+4
1530  003744  104005                           ERROR   5            ;R0 NOT $REG0+4
1531
1532  003746  026767  175604  175214          CMP     D1010,$REG0  ;CHECK FIRST WORD
1533  003754  001401                           BEQ     .+4
1534  003756  104011                           ERROR   11           ;$REG0 NOT D1010
1535  003760  026767  175574  175204          CMP     D1010+2,$REG1 ;SECOND
1536  003766  001401                           BEQ     .+4
1537  003770  104011                           ERROR   11           ;$REG1 NOT D1010+2
1538
1539  003772  170011                           SETD                 ;GO TO DOUBLE MODE
1540  003774  012700  001200                   MOV     #$REG4,R0    ;ADDRESS OF DATA+10
1541  004000  174040                           STD     AC0,-(R0)    ;GET DATA
1542  004002  022700  001170                   CMP     #$REG0,R0    ;CHECK FOR PROPER DECREMENTATION
1543  004006  001401                           BEQ     .+4
1544  004010  104005                           ERROR   5            ;R0 NOT $REG0
1545
1546  004012  012700  001610                   MOV     #AD1000,R0   ;LOAD ADDRESS OF ADDRESS OF DATA
1547  004016  173430                           CMPD    @(R0)+,AC0   ;CHECK THE DATA
1548  004020  022700  001612                   CMP     #AD1000+2,R0 ;R0 GETS INCREMENTED BY 2
1549  004024  001401                           BEQ     .+4
1550  004026  104005                           ERROR   5            ;R0 NOT AD1001+2
1551
1552  004030  170267  175430                   STFPS   FPS          ;GET STATUS
1553  004034  170000                           CFCC                 ;COPY CONDITION CODES
1554  004036  001401                           BEQ     .+4          ;EITHER CMPD FAILED OR THE
1555  004040  104011                           ERROR   11           ;LDF MODIFIED RIGHT HALF
```

```
1556                                    ;;*****************************************************************
1557                                    ;*TEST 6        TEST OF LDF, STF, CMPF WITH <=> IN ALL AC'S
1558                                    ;;*****************************************************************
1559  004042  000004                   TST6:   SCOPE
1560  004044  170127  047400                   LDFPS   #47400          ;LOAD FLOATING MODE
1561  004050  172427  140640                   LDF     #-5,AC0         ;LOAD AC0 WITH -5
1562  004054  172527  140200                   LDF     #-1,AC1         ;LOAD AC1 WITH -1
1563  004060  172627  000000                   LDF     #0,AC2          ;LOAD AC2 WITH 0
1564  004064  172727  040640                   LDF     #5,AC3          ;LOAD AC3 WITH 5
1565  004070  174004                           STF     AC0,AC4         ;LOAD AC4 WITH -5
1566  004072  174305                           STF     AC3,AC5         ;LOAD AC5 WITH 5
1567
1568  004074  174067  175070                   STF     AC0,$REG0       ;GET AC0
1569  004100  173427  140640                   CMPF    #-5,AC0         ;CHECK IT
1570  004104  170267  175354                   STFPS   FPS             ;GET STATUS
1571  004110  170000                           CFCC                    ;GET CC
1572  004112  001401                           BEQ     .+4
1573  004114  104011                           ERROR   11              ;AC0 NOT -5
1574
1575  004116  174137  001170                   STF     AC1,@#$REG0     ;GET AC1
1576  004122  173567  175042                   CMPF    $REG0,AC1       ;CHECK IT
1577  004126  170267  175332                   STFPS   FPS             ;GET STATUS
1578  004132  170000                           CFCC                    ;GET CC
1579  004134  001401                           BEQ     .+4
1580  004136  104011                           ERROR   11              ;AC1 NOT -1
1581  004140  012704  001170                   MOV     #$REG0,R4       ;POINTER TO ANSWER AREA
1582  004144  174214                           STF     AC2,(4)         ;PUT DATA INTO $REG0
1583  004146  173667  175016                   CMPF    $REG0,AC2       ;CHECK DATA
1584  004152  170267  175306                   STFPS   FPS             ;GET STATUS
1585  004156  170000                           CFCC                    ;GET CC
1586  004160  001401                           BEQ     .+4
1587  004162  104011                           ERROR   11              ;AC2 NOT 0
1588
1589  004164  174377  175442                   STF     AC3,@A$REG0     ;PUT DATA INTO $REG0
1590  004170  173767  174774                   CMPF    $REG0,AC3       ;CHECK DATA
1591  004174  170267  175264                   STFPS   FPS             ;GET STATUS
1592  004200  170000                           CFCC                    ;GET CC
1593  004202  001401                           BEQ     .+4
1594  004204  104011                           ERROR   11              ;AC3 NOT 5
1595
1596  004206  173404                           CMPF    AC4,AC0         ;CHECK AC4 FOR -5
1597  004210  170267  175250                   STFPS   FPS             ;GET STATUS
1598  004214  170000                           CFCC                    ;GET CC
1599  004216  001401                           BEQ     .+4
1600  004220  104000                           ERROR   0               ;AC0 NOT AC4
1601
1602  004222  173705                           CMPF    AC5,AC3         ;CHECK AC5 FOR 5
1603  004224  170267  175234                   STFPS   FPS             ;GET STATUS
1604  004230  170000                           CFCC                    ;GET CC
1605  004232  001401                           BEQ     .+4
1606  004234  104000                           ERROR   0               ;AC3 NOT AC5
```

```
1607                                    ;;*****************************************************************
1608                                    ;*TEST 7        TEST OF CMPF WITH DATA IN AC0-AC5
1609                                    ;;*****************************************************************
1610  004236  000004                    TST7:   SCOPE
1611  004240  170127  041000                   LDFPS   #41000          ;LOAD STATUS WITH 0
1612  004244  173405                           CMPF    AC5,AC0         ;CMP 5 TO -5
1613  004246  170267  175212                   STFPS   FPS             ;STORE FLOATING POINT STATUS
1614  004252  022767  041000  175204           CMP     #41000,FPS      ;CHECK FLOATING POINT STATUS
1615  004260  001401                           BEQ     .+4             ;BRANCH IF OK
1616  004262  104001                           ERROR   1               ;FPS NOT EQUAL TO 41000
1617
1618  004264  173704                           CMPF    AC4,AC3         ;CMP -5 TO 5
1619  004266  170267  175172                   STFPS   FPS             ;STORE FLOATING POINT STATUS
1620  004272  022767  041010  175164           CMP     #41010,FPS      ;CHECK FLOATING POINT STATUS
1621  004300  001401                           BEQ     .+4             ;BRANCH IF OK
1622  004302  104001                           ERROR   1               ;FPS NOT EQUAL TO 41010
1623
1624  004304  173767  175302                   CMPF    DBIG,AC3        ;MAKE IT OVERFLOW
1625  004310  170267  175150                   STFPS   FPS             ;GET STATUS
1626  004314  022767  041000  175142           CMP     #41000,FPS      ;CHECK FLOATING POINT STATUS
1627  004322  001401                           BEQ     .+4
1628  004324  104001                           ERROR   1               ;FPS NOT 141002
1629
```

```
                                        ;;*********************************************************************
                                        ;*TEST 10       TEST OF TSTF AND TSTD USING OLD AC0-AC5
                                        ;;*********************************************************************
004326  000004                  TST10:  SCOPE
004330  170127  040000                  LDFPS   #40000
004334  170501                          TSTF    AC1             ;TEST AC1 = -1
004336  170267  175122                  STFPS   FPS             ;GET STATUS
004342  022767  040010  175114          CMP     #40010,FPS      ;CHECK STATUS
004350  001401                          BEQ     .+4
004352  104001                          ERROR   1               ;N BIT NOT SET

004354  170505                          TSTF    AC5             ;TEST AC5 = 5
004356  170267  175102                  STFPS   FPS             ;GET STATUS
004362  022767  040000  175074          CMP     #40000,FPS      ;CHECK STATUS
004370  001401                          BEQ     .+4
004372  104001                          ERROR   1               ;NOT 0

004374  170502                          TSTF    AC2             ;TEST AC2 = 0
004376  170267  175062                  STFPS   FPS             ;GET STATUS
004402  022767  040004  175054          CMP     #40004,FPS      ;CHECK STATUS
004410  001401                          BEQ     .+4
004412  104001                          ERROR   1               ;Z BIT NOT SET

004414  170527  177777                  TSTF    #-1             ;TEST FOR THE N BIT IN LINE
004420  000401                          BR      .+4             ;SHOULD GO HERE
004422  104001                          ERROR   1               ;INCREMENTED BY 4 NOT 2
004424  170267  175034                  STFPS   FPS             ;GET STATUS
004430  022767  040010  175026          CMP     #40010,FPS      ;CHECK THE N BIT
004436  001401                          BEQ     .+4
004440  104001                          ERROR   1               ;N BIT NOT SET

004442  170011                          SETD                    ;SET DOUBLE MODE
004444  170527  000000                  TSTD    #0              ;TEST FOR Z BIT IN LINE
004450  000403                          BR      .+10            ;SHOULD GO HERE
004452  104001                          ERROR   1               ;NOT HERE
004454  104001                          ERROR   1               ;OR HERE
004456  104001                          ERROR   1               ;OR HERE
004460  170267  175000                  STFPS   FPS             ;GET STATUS
004464  022767  040204  174772          CMP     #40204,FPS      ;CHECK STATUS
004472  001401                          BEQ     .+4
004474  104001                          ERROR   1               ;Z BIT NOT SET
```

```
1671                                    ;;*****************************************************************
1672                                    ;*TEST 11       TEST OF CLRX INSTRUCTIONS
1673                                    ;;*****************************************************************
1674  004476  000004                    TST11:  SCOPE
1675  004500  170127  040200                   LDFPS   #40200          ;DOUBLE MODE
1676  004504  172467  175046                   LDD     D1010,AC0       ;LOAD A -1
1677  004510  174067  174454                   STD     AC0,$REG0       ;PUT INTO $REG0
1678  004514  170001                            SETF                    ;SET FLOATING
1679  004516  170467  174446                   CLRF    $REG0           ;CLEAR IT OUT
1680  004522  170267  174736                   STFPS   FPS             ;STORE FLOATING POINT STATUS
1681  004526  022767  040004  174730          CMP     #40004,FPS      ;CHECK FLOATING POINT STATUS
1682  004534  001401                            BEQ     .+4             ;BRANCH IF OK
1683  004536  104001                            ERROR   1               ;FPS NOT EQUAL TO 40004
1684
1685  004540  170567  174424                   TSTF    $REG0           ;TEST FOR ZERO
1686  004544  170000                            CFCC                    ;GET CC
1687  004546  001401                            BEQ     .+4
1688  004550  104011                            ERROR   11              ;AC0 NOT ZERO
1689
1690  004552  026767  175004  174414          CMP     D1010+4,$REG2   ;CHECK THIRD WORD
1691  004560  001401                            BEQ     .+4
1692  004562  104010                            ERROR   10              ;$REG2 NOT -1
1693  004564  026767  174774  174404          CMP     D1010+6,$REG3   ;CHECK FOURTH
1694  004572  001401                            BEQ     .+4
1695  004574  104010                            ERROR   10              ;$REG3 NOT 0
1696
1697  004576  170011                            SETD
1698  004600  170400                            CLRD    AC0             ;CLEAR THE REST OF AC0
1699  004602  170267  174656                   STFPS   FPS             ;STORE FLOATING POINT STATUS
1700  004606  022767  040204  174650          CMP     #40204,FPS      ;CHECK FLOATING POINT STATUS
1701  004614  001401                            BEQ     .+4             ;BRANCH IF OK
1702  004616  104001                            ERROR   1               ;FPS NOT EQUAL TO 40204
1703  004620  174067  174344                   STD     AC0,$REG0       ;STORE RESULT
1704  004624  170500                            TSTD    AC0             ;DID IT CLEAR
1705  004626  170000                            CFCC                    ;GET STATUS
1706  004630  001401                            BEQ     .+4
1707  004632  104010                            ERROR   10              ;DID NOT CLEAR
1708
1709  004634  172467  174762                   LDD     DMZERO,AC0      ;LOAD A MINUS ZERO
1710  004640  170400                            CLRD    AC0             ;CLEAR IT OUT
1711  004642  170267  174616                   STFPS   FPS             ;STORE FLOATING POINT STATUS
1712  004646  022767  040204  174610          CMP     #40204,FPS      ;CHECK FLOATING POINT STATUS
1713  004654  001401                            BEQ     .+4             ;BRANCH IF OK
1714  004656  104001                            ERROR   1               ;FPS NOT EQUAL TO 40204
1715  004660  174067  174304                   STD     AC0,$REG0       ;STORE RESULT
1716  004664  170500                            TSTD    AC0             ;CHECK IT
1717  004666  170000                            CFCC                    ;GET CC
1718  004670  001401                            BEQ     .+4
1719  004672  104010                            ERROR   10              ;DID NOT CLEAR
```

```
1720                                    ;;*****************************************************************
1721                                    ;*TEST 12        TEST OF NEGX
1722                                    ;;*****************************************************************
1723  004674  000004                    TST12:  SCOPE
1724  004676  170127  040000                    LDFPS   #40000
1725  004702  172427  140640                    LDF     #-5,AC0         ;LOAD AC0 WITH -5
1726  004706  170700                            NEGF    AC0             ;MAKE IT 5
1727  004710  170267  174550                    STFPS   FPS             ;STORE FLOATING POINT STATUS
1728  004714  022767  040000  174542            CMP     #40000,FPS      ;CHECK FLOATING POINT STATUS
1729  004722  001401                            BEQ     .+4             ;BRANCH IF OK
1730  004724  104001                            ERROR   1               ;FPS NOT EQUAL TO 40000
1731
1732  004726  174067  174236                    STF     AC0,$REG0       ;GET THE RESULT
1733  004732  173427  040640                    CMPF    #5,AC0          ;CHECK THE RESULT
1734  004736  170000                            CFCC                    ;GET CC
1735  004740  001401                            BEQ     .+4
1736  004742  104011                            ERROR   11              ;RESULT NOT 5
1737
1738  004744  170767  174220                    NEGF    $REG0           ;MAKE IT -5
1739  004750  170267  174510                    STFPS   FPS             ;STORE FLOATING POINT STATUS
1740  004754  022767  040010  174502            CMP     #40010,FPS      ;CHECK FLOATING POINT STATUS
1741  004762  001401                            BEQ     .+4             ;BRANCH IF OK
1742  004764  104001                            ERROR   1               ;FPS NOT EQUAL TO 40010
1743
1744  004766  022767  140640  174174            CMP     #140640,$REG0   ;CHECK THE RESULT
1745  004774  001401                            BEQ     .+4
1746  004776  104011                            ERROR   11              ;RESULT NOT -5
1747
1748  005000  005767  174166                    TST     $REG1           ;REST 0?
1749  005004  001401                            BEQ     .+4             ;SKIP IF OK
1750  005006  104011                            ERROR   11
1751
1752  005010  170127  047400                    LDFPS   #47400          ;TURN ON INTERUPTS
1753  005014  170400                            CLRF    AC0             ;CLEAR AC0
1754  005016  170700                            NEGF    AC0             ;NEGATE IT
1755  005020  170267  174440                    STFPS   FPS             ;STORE FLOATING POINT STATUS
1756  005024  022767  047404  174432            CMP     #47404,FPS      ;CHECK FLOATING POINT STATUS
1757  005032  001401                            BEQ     .+4             ;BRANCH IF OK
1758  005034  104001                            ERROR   1               ;FPS NOT EQUAL TO 47404
1759
1760  005036  174067  174126                    STF     AC0,$REG0       ;GET RESULT
1761  005042  170500                            TSTF    AC0             ;CHECK IT
1762  005044  170000                            CFCC                    ;GET CC
1763  005046  001401                            BEQ     .+4
1764  005050  104011                            ERROR   11              ;RESULT NOT 0
1765
1766  005052  170011                            SETD                    ;SET DOUBLE MODE
1767  005054  170400                            CLRD    AC0             ;CLEAR AC0
1768  005056  170700                            NEGD    AC0             ;NEGATE AC0
1769  005060  170267  174400                    STFPS   FPS             ;STORE FLOATING POINT STATUS
1770  005064  022767  047604  174372            CMP     #47604,FPS      ;CHECK FLOATING POINT STATUS
1771  005072  001401                            BEQ     .+4             ;BRANCH IF OK
1772  005074  104001                            ERROR   1               ;FPS NOT EQUAL TO 47604
1773
1774  005076  174067  174066                    STF     AC0,$REG0       ;GET RESULT
1775  005102  170500                            TSTF    AC0             ;TEST RESULT
```

```
1776  005104  170000                    CFCC                    ;GET CC
1777  005106  001401                    BEQ     .+4
1778  005110  104011                    ERROR   11              ;RESULT NOT 0
1779
```

```
1780                                    ;;*****************************************************************
1781                                    ;*TEST 13        TEST OF ABSX
1782                                    ;;*****************************************************************
1783  005112  000004                    TST13:  SCOPE
1784  005114  170127  040000                    LDFPS   #40000
1785  005120  172427  140200                    LDF     #-1,AC0         ;LOAD A -1
1786  005124  174067  174040                    STF     AC0,$REG0       ;GET IT
1787  005130  170667  174034                    ABSF    $REG0           ;MAKE IT A 1
1788  005134  170267  174324                    STFPS   FPS             ;STORE FLOATING POINT STATUS
1789  005140  022767  040000  174316            CMP     #40000,FPS      ;CHECK FLOATING POINT STATUS
1790  005146  001401                            BEQ     .+4             ;BRANCH IF OK
1791  005150  104001                            ERROR   1               ;FPS NOT EQUAL TO 40000
1792
1793  005152  022767  040200  174010            CMP     #40200,$REG0    ;CHECK FOR A 1
1794  005160  001401                            BEQ     .+4
1795  005162  104011                            ERROR   11              ;RESULT NOT 1
1796
1797  005164  005767  174002                    TST     $REG1           ;REST 0?
1798  005170  001401                            BEQ     .+4             ;SKIP IF OK
1799  005172  104011                            ERROR   11
1800
1801  005174  170600                            ABSF    AC0             ;ABS IT AGAIN
1802  005176  170267  174262                    STFPS   FPS             ;GET STATUS
1803  005202  174067  173762                    STF     AC0,$REG0       ;STORE ANSWER
1804  005206  173427  040200                    CMPF    #1,AC0          ;CHECK FOR A 1
1805  005212  170000                            CFCC                    ;GET CC
1806  005214  001401                            BEQ     .+4
1807  005216  104011                            ERROR   11              ;RESULT NOT 1
1808
1809  005220  172467  174376                    LDF     DMZERO,AC0      ;LOAD A MINUS ZERO
1810  005224  170600                            ABSF    AC0             ;ABS IT
1811  005226  170267  174232                    STFPS   FPS             ;STORE FLOATING POINT STATUS
1812  005232  022767  040004  174224            CMP     #40004,FPS      ;CHECK FLOATING POINT STATUS
1813  005240  001401                            BEQ     .+4             ;BRANCH IF OK
1814  005242  104001                            ERROR   1               ;FPS NOT EQUAL TO 40004
1815
1816  005244  174067  173720                    STF     AC0,$REG0       ;GET RESULT
1817  005250  170500                            TSTF    AC0             ;TEST FOR 0
1818  005252  170000                            CFCC                    ;GET CC
1819  005254  001401                            BEQ     .+4
1820  005256  104011                            ERROR   11              ;RESULT NOT 0
1821
1822  005260  170011                            SETD                    ;SET DOUBLE MODE
1823  005262  172467  174270                    LDD     D1010,AC0       ;LOAD -1,0,-1,0
1824  005266  170600                            ABSD    AC0             ;ABS IT
1825  005270  170267  174170                    STFPS   FPS             ;STORE FLOATING POINT STATUS
1826  005274  022767  040200  174162            CMP     #40200,FPS      ;CHECK FLOATING POINT STATUS
1827  005302  001401                            BEQ     .+4             ;BRANCH IF OK
1828  005304  104001                            ERROR   1               ;FPS NOT EQUAL TO 40200
1829  005306  174067  173656                    STD     AC0,$REG0       ;GET RESULT
1830  005312  173467  174274                    CMPD    DBIG,AC0        ;CHECK THE RESULT
1831  005316  170000                            CFCC                    ;GET CC
1832  005320  001401                            BEQ     .+4
1833  005322  104010                            ERROR   10              ;RESULT NOT 77777,0,-1,0
1834
```

```
1835                                    ;;*****************************************************************
1836                                    ;*TEST 14       TEST OF LDEXP & STEXP
1837                                    ;;*****************************************************************
1838  005324  000004                   TST14:  SCOPE
1839  005326  170127  040200                   LDFPS   #40200          ;SET DOUBLE MODE
1840  005332  172467  174220                   LDD     D1010,AC0       ;LOAD A -1,0,-1,0
1841  005336  176427  177600                   LDEXP   #-200,AC0       ;CLEAR THE EXPONENT
1842  005342  170267  174116                   STFPS   FPS             ;STORE FLOATING POINT STATUS
1843  005346  022767  040204  174110           CMP     #40204,FPS      ;CHECK FLOATING POINT STATUS
1844  005354  001401                           BEQ     .+4             ;BRANCH IF OK
1845  005356  104001                           ERROR   1               ;FPS NOT EQUAL TO 40204
1846
1847  005360  174067  173604                   STD     AC0,$REG0       ;GET THE RESULT
1848  005364  170500                           TSTD    AC0             ;IS IT 0
1849  005366  170000                           CFCC
1850  005370  001401                           BEQ     .+4
1851  005372  104010                           ERROR   10              ;AC0 NOT 0
1852
1853  005374  175067  173570                   STEXP   AC0,$REG0       ;GET THE RESULT
1854  005400  013700  177776                   MOV     @#PS,R0         ;GET PS BITS
1855  005404  042700  177760                   BIC     #177760,R0      ;CLEAR JUNK
1856  005410  022700  000010                   CMP     #10,R0          ;IS IT OK?
1857  005414  001401                           BEQ     .+4             ;SKIP IF OK
1858  005416  104004                           ERROR   4               ;PS IS WRONG
1859
1860  005420  022767  177600  173542           CMP     #-200,$REG0     ;CHECK IT
1861  005426  001401                           BEQ     .+4
1862  005430  104012                           ERROR   12              ;EXPONENT NOT 0
1863
1864  005432  170001                           SETF                    ;SET FLOATING MODE
1865  005434  172467  174120                   LDF     D0101,AC0       ;LOAD A 0,-1
1866  005440  176427  000200                   LDEXP   #200,AC0        ;SET EXPONENT TO -1
1867  005444  170267  174014                   STFPS   FPS             ;STORE FLOATING POINT STATUS
1868  005450  022767  040006  174006           CMP     #40006,FPS      ;CHECK FLOATING POINT STATUS
1869  005456  001401                           BEQ     .+4             ;BRANCH IF OK
1870  005460  104001                           ERROR   1               ;FPS NOT EQUAL TO 40006
1871
1872  005462  174067  173502                   STF     AC0,$REG0       ;SAVE RESULT
1873  005466  005767  173476                   TST     $REG0           ;CHECK FIRST WORD
1874  005472  001401                           BEQ     .+4
1875  005474  104011                           ERROR   11              ;$REG0 NOT 0
1876
1877  005476  005767  173470                   TST     $REG1           ;CHECK SECOND WORD
1878  005502  001401                           BEQ     .+4
1879  005504  104011                           ERROR   11              ;$REG1 NOT ZERO
1880
1881  005506  175067  173456                   STEXP   AC0,$REG0       ;GET THE EXPONENT BACK
1882  005512  022767  177600  173450           CMP     #-200,$REG0     ;CHECK IT
1883  005520  001401                           BEQ     .+4
1884  005522  104012                           ERROR   12              ;EXPONENT NOT -200
1885
1886  005524  176527  000052                   LDEXP   #52,AC1         ;LOAD ALT 1'S
1887  005530  175100                           STEXP   AC1,R0          ;GET THEM BACK
1888  005532  022700  000052                   CMP     #52,R0          ;OK?
1889  005536  001401                           BEQ     .+4
1890  005540  104013                           ERROR   13              ;EXP NOT 252
```

```
1891
1892  005542  176627  000025          LDEXP   #25,AC2         ;LOAD OTHER ALT 1'S
1893  005546  175200                  STEXP   AC2,R0          ;GET IT BACK
1894  005550  022700  000025          CMP     #25,R0          ;CHECK IT
1895  005554  001401                  BEQ     .+4
1896  005556  104013                  ERROR   13              ;EXP NOT 125
1897
```

```
1898                                    ;;*****************************************************************
1899                                    ;*TEST 15       TEST OF ADDF & SUBF
1900                                    ;;*****************************************************************
1901  005560  000004                   TST15:  SCOPE
1902  005562  170127  047400                   LDFPS   #47400          ;LOAD FLOATING MODE
1903  005566  172427  040600                   LDF     #4,AC0          ;LOAD A 4
1904  005572  172027  040400                   ADDF    #2,AC0          ;ADD A 2
1905  005576  170267  173662                   STFPS   FPS             ;STORE FLOATING POINT STATUS
1906  005602  022767  047400  173654           CMP     #47400,FPS      ;CHECK FLOATING POINT STATUS
1907  005610  001401                           BEQ     .+4             ;BRANCH IF OK
1908  005612  104001                           ERROR   1               ;FPS NOT EQUAL TO 47400
1909
1910  005614  174067  173350                   STF     AC0,$REG0       ;STORE RESULT
1911  005620  173427  040700                   CMPF    #6,AC0          ;CHECK RESULT
1912  005624  170000                           CFCC                    ;GET CC
1913  005626  001401                           BEQ     .+4
1914  005630  104011                           ERROR   11              ;RESULT NOT 6
1915
1916  005632  173027  041020                   SUBF    #9,AC0          ;SUBTRACT 9 FROM 6
1917  005636  170267  173622                   STFPS   FPS             ;STORE FLOATING POINT STATUS
1918  005642  022767  047410  173614           CMP     #47410,FPS      ;CHECK FLOATING POINT STATUS
1919  005650  001401                           BEQ     .+4             ;BRANCH IF OK
1920  005652  104001                           ERROR   1               ;FPS NOT EQUAL TO 47410
1921
1922  005654  174067  173310                   STF     AC0,$REG0       ;STORE RESULT IN $REG0
1923  005660  173427  140500                   CMPF    #-3,AC0         ;CHECK RESULT
1924  005664  170000                           CFCC                    ;GET CC
1925  005666  001401                           BEQ     .+4
1926  005670  104011                           ERROR   11              ;6 - 9 NOT -3?
1927
1928
```

```
                                ;;*****************************************************************
                                ;*TEST 16        TEST OF ADDD AND SUBD
                                ;;*****************************************************************
005672  000004                  TST16:  SCOPE
005674  170127  047600                  LDFPS   #47600          ;SET DOUBLE MODE
005700  172527  141400                  LDD     #-32.,AC1       ;LOAD A -32.
005704  172127  040640                  ADDD    #5,AC1          ;ADD 5 TO -32.
005710  170267  173550                  STFPS   FPS             ;STORE FLOATING POINT STATUS
005714  022767  047610  173542          CMP     #47610,FPS      ;CHECK FLOATING POINT STATUS
005722  001401                          BEQ     .+4             ;BRANCH IF OK
005724  104001                          ERROR   1               ;FPS NOT EQUAL TO 47610

005726  174177  173700                  STD     AC1,@A$REGO     ;GET RESULT
005732  173527  141330                  CMPD    #-27.,AC1       ;CHECK RESULT
005736  170000                          CFCC                    ;GET CC
005740  001401                          BEQ     .+4
005742  104010                          ERROR   10              ;-32.+5 NOT -27

005744  172667  173664                  LDD     DALTA,AC2       ;LOAD ALT 1'S
005750  172267  173670                  ADDD    DALTB,AC2       ;ADD OTHER 1'S
005754  170267  173504                  STFPS   FPS             ;STORE FLOATING POINT STATUS
005760  022767  047600  173476          CMP     #47600,FPS      ;CHECK FLOATING POINT STATUS
005766  001401                          BEQ     .+4             ;BRANCH IF OK
005770  104001                          ERROR   1               ;FPS NOT EQUAL TO 47600

005772  174277  173634                  STD     AC2,@A$REGO     ;GET RESULT
005776  173627  040500                  CMPD    #3,AC2          ;CHECK RESULT
006002  170000                          CFCC                    ;GET CC
006004  001401                          BEQ     .+4
006006  104010                          ERROR   10              ;$REGO NOT DALTAN

006010  173267  173620                  SUBD    DALTA,AC2       ;SUBTRACT IT BACK
006014  170267  173444                  STFPS   FPS             ;STORE FLOATING POINT STATUS
006020  022767  047600  173436          CMP     #47600,FPS      ;CHECK FLOATING POINT STATUS
006026  001401                          BEQ     .+4             ;BRANCH IF OK
006030  104001                          ERROR   1               ;FPS NOT EQUAL TO 47600

006032  174267  173132                  STD     AC2,$REGO       ;GET THE RESULT
006036  173667  173612                  CMPD    DALTC,AC2       ;CHECK IT
006042  170000                          CFCC                    ;GET CC
006044  001401                          BEQ     .+4
006046  104010                          ERROR   10              ;$REGO NOT DALTA
```

```
                                ;;*****************************************************************
                                ;*TEST 17       TEST OF MULF AND DIVF
                                ;;*****************************************************************
006050  000004                  TST17:  SCOPE
006052  170127  047400                  LDFPS   #47400          ;LOAD FLOATING MODE
006056  172727  040740                  LDF     #7,AC3          ;LOAD A7
006062  171327  140500                  MULF    #-3,AC3         ;X -3
006066  170267  173372                  STFPS   FPS             ;STORE FLOATING POINT STATUS
006072  022767  047410  173364          CMP     #47410,FPS      ;CHECK FLOATING POINT STATUS
006100  001401                          BEQ     .+4             ;BRANCH IF OK
006102  104001                          ERROR   1               ;FPS NOT EQUAL TO 47410

006104  174367  173060                  STF     AC3,$REG0       ;GET RESULT
006110  173727  141250                  CMPF    #-21.,AC3       ;CHECK RESULT
006114  170000                          CFCC                    ;GET CC
006116  001401                          BEQ     .+4
006120  104011                          ERROR   11              ;X -3 NOT -21.

006122  174727  140740                  DIVF    #-7,AC3         ;DIVIDE BY -3
006126  170267  173332                  STFPS   FPS             ;STORE FLOATING POINT STATUS
006132  022767  047400  173324          CMP     #47400,FPS      ;CHECK FLOATING POINT STATUS
006140  001401                          BEQ     .+4             ;BRANCH IF OK
006142  104001                          ERROR   1               ;FPS NOT EQUAL TO 47400

006144  174367  173020                  STF     AC3,$REG0       ;GET RESULT
006150  173727  040500                  CMPF    #3,AC3          ;CHECK RESULT
006154  170000                          CFCC                    ;GET CC
006156  001401                          BEQ     .+4
006160  104011                          ERROR   11              ;-21. / -7 NOT 3
```

```
2000                                    ;;*****************************************************************
2001                                    ;*TEST 20        TEST OF MULD AND DIVD
2002                                    ;;*****************************************************************
2003  006162  000004                    TST20:  SCOPE
2004  006164  170127  047600                    LDFPS   #47600          ;LOAD DOUBLE MODE
2005  006170  172427  140640                    LDD     #-5,AC0         ;LOAD A -5
2006  006174  171027  140500                    MULD    #-3,AC0         ;MUL BY -3
2007  006200  170267  173260                    STFPS   FPS             ;STORE FLOATING POINT STATUS
2008  006204  022767  047600  173252            CMP     #47600,FPS      ;CHECK FLOATING POINT STATUS
2009  006212  001401                            BEQ     .+4             ;BRANCH IF OK
2010  006214  104001                            ERROR   1               ;FPS NOT EQUAL TO 47600
2011
2012  006216  174067  172746                    STD     AC0,$REG0       ;GET RESULT
2013  006222  173427  041160                    CMPD    #15.,AC0        ;CHECK RESULT
2014  006226  170000                            CFCC                    ;GET CC
2015  006230  001401                            BEQ     .+4
2016  006232  104010                            ERROR   10              ;-5 X -3 NOT +15.
2017
2018  006234  174427  140400                    DIVD    #-2,AC0         ;15. / -2
2019  006240  170267  173220                    STFPS   FPS             ;STORE FLOATING POINT STATUS
2020  006244  022767  047610  173212            CMP     #47610,FPS      ;CHECK FLOATING POINT STATUS
2021  006252  001401                            BEQ     .+4             ;BRANCH IF OK
2022  006254  104001                            ERROR   1               ;FPS NOT EQUAL TO 47610
2023
2024  006256  174067  172706                    STD     AC0,$REG0       ;STORE RESULT
2025  006262  173427  140760                    CMPD    #-7.5,AC0       ;CHECK RESULT
2026  006266  170000                            CFCC                    ;GET CC
2027  006270  001401                            BEQ     .+4
2028  006272  104010                            ERROR   10              ;15. / -2 NOT -7.5
```

```
                                ;;*****************************************************************
                                ;*TEST 21        TEST OF LDCFD,LDCDF
                                ;;*****************************************************************
006274  000004                  TST21:  SCOPE
006276  170127  047600                  LDFPS   #47600          ;SET DOUBLE MODE
006302  172477  173276                  LDD     @AD0101,AC0     ;LOAD A 0,-1,0,-1
006306  012700  001556                  MOV     #D1010,R0       ;GET ADDRESS OF DATA
006312  177420                          LDCFD   (R0)+,AC0       ;LOAD A -1,0,0,0
006314  022700  001562                  CMP     #D1010+4,R0     ;INC BY 4?
006320  001401                          BEQ     .+4
006322  104005                          ERROR   5               ;R0 NOT #D1010+4
006324  170267  173134                  STFPS   FPS             ;STORE FLOATING POINT STATUS
006330  022767  047610  173126          CMP     #47610,FPS      ;CHECK FLOATING POINT STATUS
006336  001401                          BEQ     .+4             ;BRANCH IF OK
006340  104001                          ERROR   1               ;FPS NOT EQUAL TO 47610
006342  174067  172622                  STD     AC0,$REG0       ;GET ANSWER
006346  173467  173354                  CMPD    D1000,AC0       ;CHECK RESULT
006352  170000                          CFCC                    ;GET CC
006354  001401                          BEQ     .+4
006356  104010                          ERROR   10              ;AC0 NOT -1,0,0,0
006360  172567  173174                  LDD     D0101,AC1       ;LOAD A 0,-1,0,-1
006364  170001                          SETF                    ;SET FLOATING MODE
006366  012700  001566                  MOV     #D1010+10,R0    ;GET ADDRESS OF DATA +10
006372  177540                          LDCDF   -(R0),AC1       ;LOAD A -1,0
006374  022700  001556                  CMP     #D1010,R0       ;ADDRESS DECREMENT BY 10?
006400  001401                          BEQ     .+4
006402  104005                          ERROR   5               ;R0 NOT #D1010
006404  170267  173054                  STFPS   FPS             ;STORE FLOATING POINT STATUS
006410  022767  047410  173046          CMP     #47410,FPS      ;CHECK FLOATING POINT STATUS
006416  001401                          BEQ     .+4             ;BRANCH IF OK
006420  104001                          ERROR   1               ;FPS NOT EQUAL TO 47410
006422  170011                          SETD
006424  174167  172540                  STD     AC1,$REG0       ;GET RESULT
006430  173567  173140                  CMPD    WEIRD,AC1       ;CHECK RESULT
006434  170000                          CFCC
006436  001401                          BEQ     .+4
006440  104010                          ERROR   10              ;RESULT NOT -1,1,0,-1
006442  170127  047440                  LDFPS   #47440          ;SET DOUBLE AND TRUNCATE MODES
006446  177567  173104                  LDCDF   D1010,AC1       ;LOAD IT
006452  174167  172512                  STD     AC1,$REG0       ;GET RESULT
006456  173567  173104                  CMPD    D1001,AC1       ;CHECK RESULT
006462  170267  172776                  STFPS   FPS             ;GET STATUS
006466  170000                          CFCC                    ;GET CC
006470  001401                          BEQ     .+4
006472  104010                          ERROR   10              ;AC1 NOT -1,0,0,1
006474  170127  047400                  LDFPS   #47400          ;SET ROUND AND FLOATING MODES
006500  177567  173052                  LDCDF   D1010,AC1       ;LOAD A -1.0
006504  174167  172460                  STD     AC1,$REG0       ;GET THE RESULT
006510  170267  172750                  STFPS   FPS             ;GET STATUS
```

```
 2085  006514  022767  000001  172450          CMP     #1,$REG1        ;CHECK WORD 2 FOR A 1
 2086  006522  001401                          BEQ     .+4
 2087  006524  104010                          ERROR   10              ;LDCDF DID NOT ROUND PROPERLY
 2088
```

```
                                ;;*****************************************************************
                                ;#TEST 22       TEST OF STCFD,STCDF
                                ;;*****************************************************************
006526  000004                  TST22:  SCOPE
006530  170127  040200                  LDFPS   #40200          ;SET DOUBLE MODE
006534  172667  173016                  LDD     D1010,AC2       ;LOAD A -1,0,-1,0
006540  170001                          SETF                    ;SET FLOATING MODE
006542  176202                          STCDF   AC2,AC2         ;CLEAR RIGHT HALF
006544  170011                          SETD                    ;SET DOUBLE MODE
006546  174267  172416                  STD     AC2,$REG0       ;GET RESULT
006552  173667  173150                  CMPD    D1000,AC2       ;IS IT -1,0,0,0
006556  170000                          CFCC                    ;GET CC
006560  001401                          BEQ     .+4
006562  104010                          ERROR   10              ;AC1 NOT -1,0,0,0

006564  172767  173154                  LDD     D0111,AC3       ;LOAD A 0,-1,0,-1
006570  176303                          STCFD   AC3,AC3         ;CLEAR OUT RIGHT HALF!?
006572  174367  172372                  STD     AC3,$REG0       ;GET RESULT
006576  173767  173134                  CMPD    D0100X,AC3      ;CHECK RESULT
006602  170000                          CFCC                    ;GET CC
006604  001401                          BEQ     .+4
006606  104010                          ERROR   10              ;$REG2 NOT 100,0,0 (ROUND)

006610  172467  173130          LDD     D0111,AC0       ;LOAD 0,-1,0,-1
006614  170127  000040          LDFPS   #40             ;FLOATING AND TRUNCATE MODES
006620  176000                  STCFD   AC0,AC0         ;CLEAN RIGHT HALF
006622  170011                  SETD                    ;SET DOUBLE
006624  174067  172340          STD     AC0,$REG0       ;GET RESULT
006630  173467  173070          CMPD    D0100,AC0       ;CHECK IT
006634  170000                  CFCC                    ;GET CC
006636  001401                  BEQ     .+4
006640  104010                  ERROR   10              ;AC0 NOT 0,-1,0,0 (TRUNCATE)
```

```
                                ;;*****************************************************************
                                ;*TEST 23       TEST OF LDCIF,LDCID,STCFI,STCDI
                                ;;*****************************************************************
006642  000004                  TST23:  SCOPE
006644  170127  047400                  LDFPS   #47400          ;FLOATING MODE
006650  177027  000005                  LDCIF   #5,AC0          ;STORE A 5
006654  170267  172604                  STFPS   FPS             ;STORE FLOATING POINT STATUS
006660  022767  047400  172576          CMP     #47400,FPS      ;CHECK FLOATING POINT STATUS
006666  001401                          BEQ     .+4             ;BRANCH IF OK
006670  104001                          ERROR   1               ;FPS NOT EQUAL TO 47400

006672  174067  172272                  STF     AC0,$REG0       ;GET THE RESULT
006676  173427  040640                  CMPF    #5,AC0          ;CHECK IT
006702  170000                          CFCC                    ;GET CC
006704  001401                          BEQ     .+4
006706  104011                          ERROR   11              ;AC0 NOT 5.0

006710  175400                          STCFI   AC0,R0          ;CONVERT IT BACK
006712  022700  000005                  CMP     #5,R0           ;CHECK RESULT
006716  001401                          BEQ     .+4
006720  104013                          ERROR   13              ;R0 NOT 5

006722  170011                          SETD                    ;SET DOUBLE MODE
006724  172567  172630                  LDD     D0101,AC1       ;LOAD JUNK
006730  012700  001556                  MOV     #D1010,R0       ;LOAD ADDRESS OF DATA
006734  177120                          LDCID   (R0)+,AC1       ;CONVERT TO -1.0
006736  022700  001560                  CMP     #D1010+2,R0     ;CHECK ADDRESS
006742  001401                          BEQ     .+4
006744  104005                          ERROR   5               ;R0 NOT #D1010+2

006746  170267  172512                  STFPS   FPS             ;STORE FLOATING POINT STATUS
006752  022767  047610  172504          CMP     #47610,FPS      ;CHECK FLOATING POINT STATUS
006760  001401                          BEQ     .+4             ;BRANCH IF OK
006762  104001                          ERROR   1               ;FPS NOT EQUAL TO 47610

006764  174167  172200                  STD     AC1,$REG0       ;GET RESULT
006770  173527  140200                  CMPD    #-1,AC1         ;CHECK RESULT
006774  170000                          CFCC                    ;GET CC
006776  001401                          BEQ     .+4
007000  104010                          ERROR   10              ;AC1 NOT -1.0

007002  175500                          STCDI   AC1,R0          ;CONVERT IT BACK
007004  022700  177777                  CMP     #-1,R0          ;CHECK RESULT
007010  001401                          BEQ     .+4
007012  104013                          ERROR   13              ;R0 NOT -1

007014  170001                          SETF                    ;SET FLOATING MODE
007016  177227  054321                  LDCIF   #54321,AC2      ;LOAD 54321
007022  174267  172142                  STF     AC2,$REG0       ;GET RESULT
007026  170267  172432                  STFPS   FPS             ;GET STATUS
007032  173667  172724                  CMPF    F5T01,AC2       ;CHECK IT
007036  170000                          CFCC                    ;CHECK CC
007040  001401                          BEQ     .+4
007042  104011                          ERROR   11              ;AC2 NOT 54321.

007044  052737  000017  177776          BIS     #17,@#PS        ;SET PS
```

```
2178  007052  175667  172112          STCFI   AC2,$REGO       ;CONVERT IT BACK
2179  007056  013700  177776          MOV     @#PS,R0         ;GET PS
2180  007062  042700  177760          BIC     #177760,R0      ;CLEAR JUNK
2181  007066  001401                  BEQ     .+4             ;SKIP IF OK
2182  007070  104004                  ERROR   4               ;PS NOT 0
2183  007072  170267  172366          STFPS   FPS             ;GET STATUS
2184
2185  007076  022767  054321  172064  CMP     #54321,$REGO    ;CHECK RESULT
2186  007104  001401                  BEQ     .+4
2187  007106  104012                  ERROR   12              ;$REGO NOT 54321
2188
```

```
                                        ;;*****************************************************************
                                        ;*TEST 24       TEST OF LDCLF,LDCLD,STCFL,STCDL
                                        ;;*****************************************************************
007110  000004                  TST24:  SCOPE
007112  170127  047500                  LDFPS   #47500          ;FLOATING AND LONG MODES
007116  177327  177773                  LDCLF   #-5,AC3         ;LOAD A -5
007122  000401                          BR      .+4
007124  104000                          ERROR   0               ;LDCLF INCREMENTED BY 4 NOT 2

007126  170267  172332                  STFPS   FPS             ;STORE FLOATING POINT STATUS
007132  022767  047510  172324          CMP     #47510,FPS      ;CHECK FLOATING POINT STATUS
007140  001401                          BEQ     .+4             ;BRANCH IF OK
007142  104001                          ERROR   1               ;FPS NOT EQUAL TO 47510

007144  174367  172020                  STF     AC3,$REGO       ;GET THE RESULT
007150  173727                          CMPF    (7)+,AC3        ;CHECK IT
007152  144640                          144640
007154  170000                          CFCC                    ;GET CC
007156  001401                          BEQ     .+4
007160  104011                          ERROR   11              ;AC3 NOT -5

007162  012700  001632                  MOV     #A$REGO,R0      ;SET UP ADDRESS OF ADDRESS
007166  175730                          STCFL   AC3,@(R0)+      ;STORE IN $REGO
007170  022700  001634                  CMP     #A$REGO+2,R0    ;CHECK ADDRESS
007174  001401                          BEQ     .+4
007176  104005                          ERROR   5               ;ADDRESS IN FPU NOT A$REGO+2

007200  170267  172260                  STFPS   FPS             ;GET STATUS
007204  022767  177773  171756          CMP     #-5,$REGO       ;CHECK LEFT HALF
007212  001401                          BEQ     .+4
007214  104011                          ERROR   11              ;LEFT NOT -5

007216  005767  171750                  TST     $REG1           ;CHECK RIGHT HALF OF RESULT
007222  001401                          BEQ     .+4
007224  104011                          ERROR   11              ;$REG1 NOT 0

007226  170011                          SETD                    ;SET DOUBLE MODE
007230  177067  172520                  LDCLD   D5T01,AC0       ;LOAD WEIRD NUMBER
007234  170267  172224                  STFPS   FPS             ;STORE FLOATING POINT STATUS
007240  022767  047700  172216          CMP     #47700,FPS      ;CHECK FLOATING POINT STATUS
007246  001401                          BEQ     .+4             ;BRANCH IF OK
007250  104001                          ERROR   1               ;FPS NOT EQUAL TO 47700

007252  174067  171712                  STD     AC0,$REGO       ;GET RESULT
007256  173467  172500                  CMPD    F5T01,AC0       ;CHECK IT
007262  170000                          CFCC                    ;GET CC
007264  001401                          BEQ     .+4
007266  104010                          ERROR   10              ;AC0 NOT F5T01

007270  012700  001174                  MOV     #$REG2,R0
007274  175440                          STCDL   AC0,-(R0)       ;GET IT BACK
007276  022700  001170                  CMP     #$REGO,R0       ;CHECK ADDRESS
007302  001401                          BEQ     .+4
007304  104005                          ERROR   5               ;R0 NOT #$REGO

007306  170267  172152                  STFPS   FPS             ;STORE FLOATING POINT STATUS
```

```
2245  007312  022767  047700  172144          CMP     #47700,FPS      ;CHECK FLOATING POINT STATUS
2246  007320  001401                          BEQ     .+4             ;BRANCH IF OK
2247  007322  104001                          ERROR   1               ;FPS NOT EQUAL TO 47700
2248
2249  007324  170267  172134                  STFPS   FPS             ;GET STATUS
2250  007330  026767  172420  171632          CMP     D5T01,$REG0     ;CHECK LEFT
2251  007336  001401                          BEQ     .+4
2252  007340  104011                          ERROR   11              ;$REG0 NOT
2253
2254  007342  026767  172410  171622          CMP     D5T01+2,$REG1   ;CHECK RIGHT
2255  007350  001401                          BEQ     .+4
2256  007352  104011                          ERROR   11              ;$REG1 NOT
2257
```

```
                                        ;;*********************************************************************
                                        ;*TEST 25        TEST OF MODF AND MODD
                                        ;;*********************************************************************
007354  000004                  TST25:  SCOPE
007356  170127  047400                  LDFPS   #47400          ;SET FLOATING MODE
007362  172467  172260                  LDF     DALTB+2,AC0     ;LOAD 52525,52525 INTO AC0
007366  171427  040200                  MODF    #1,AC0          ;MODF BY A 1 *MGT*
007372  170267  172066                  STFPS   FPS             ;STORE FLOATING POINT STATUS
007376  022767  047404  172060          CMP     #47404,FPS      ;CHECK FLOATING POINT STATUS
007404  001401                          BEQ     .+4             ;BRANCH IF OK
007406  104001                          ERROR   1               ;FPS NOT EQUAL TO 47404

007410  174067  171554                  STF     AC0,$REG0       ;GET RESULT
007414  170500                          TSTF    AC0             ;CHECK FRACTION
007416  170000                          CFCC
007420  001401                          BEQ     .+4
007422  104011                          ERROR   11              ;FRACTION NOT 0

007424  174167  171540                  STF     AC1,$REG0       ;GET IT
007430  173567  172212                  CMPF    DALTB+2,AC1     ;CHECK INTEGER
007434  170000                          CFCC
007436  001401                          BEQ     .+4
007440  104011                          ERROR   11              ;INTEGER IS NOT 52525,52525

007442  172627  040200                  LDF     #1,AC2          ;LOAD A 1
007446  171667  172242                  MODF    D20,AC2         ;MOD BY 20000,0
007452  170267  172006                  STFPS   FPS             ;STORE FLOATING POINT STATUS
007456  022767  047400  172000          CMP     #47400,FPS      ;CHECK FLOATING POINT STATUS
007464  001401                          BEQ     .+4             ;BRANCH IF OK
007466  104001                          ERROR   1               ;FPS NOT EQUAL TO 47400

007470  174267  171474                  STF     AC2,$REG0       ;GET IT
007474  173667  172214                  CMPF    D20,AC2         ;CHECK FRACT
007500  170000                          CFCC
007502  001401                          BEQ     .+4
007504  104011                          ERROR   11              ;RESULT NOT 20000,0

007506  174367  171456                  STF     AC3,$REG0       ;GET INT
007512  170503                          TSTF    AC3             ;CHECK FOR 0
007514  170000                          CFCC
007516  001401                          BEQ     .+4
007520  104011                          ERROR   11              ;RESULT NOT 0

007522  170011                          SETD                    ;SET DOUBLE MODE
007524  172467  172154                  LDD     D46,AC0         ;LOAD A 40600,0,0,0
007530  171467  172140                  MODD    D37,AC0         ;MOD BY 37400,0,0,0
007534  174067  171430                  STD     AC0,$REG0       ;STORE ANSWER
007540  173467  172120                  CMPD    D40,AC0         ;CHECK FOR 40000,0,0,0
007544  170000                          CFCC
007546  001401                          BEQ     .+4
007550  104010                          ERROR   10              ;RESULT NOT 40000,0,0,0

007552  174367  171412                  STD     AC3,$REG0       ;GET THE RESULT
007556  170503                          TSTD    AC3             ;CHECK FOR 0
007560  170000                          CFCC
007562  001401                          BEQ     .+4
```

```
 2314  007564  104010                        ERROR   10              ;RESULT NOT 0
 2315
```

```
                                        .SBTTL  ERROR CONDITIONS (STST)

                                ;;*******************************************************************
                                ;*TEST 26        LDD OF -0
                                ;;*******************************************************************
007566  000004                  TST26:  SCOPE
007570  170127  047600                  LDFPS   #47600          ;DOUBLE MODE
007574  172467  172064                  LDD     D40,AC0         ;LOAD DUMMY DATA
007600  172467  172016          1$:     LDD     DMZERO,AC0      ;LOAD A -0
007604  170267  171654                  STFPS   FPS             ;STORE FLOATING POINT STATUS
007610  170367  171652                  STST    FEC             ;STORE EXCEPTION CODES
007614  022767  147614  171642          CMP     #147614,FPS     ;CHECK FLOATING POINT STATUS
007622  001401                          BEQ     .+4             ;BRANCH IF OK
007624  104001                          ERROR   1               ;FPS NOT EQUAL TO 147614
007626  022767  000014  171632          CMP     #14,FEC         ;CHECK FLOATING EXCEPTION CODE
007634  001401                          BEQ     .+4             ;BRANCH IF OK
007636  104001                          ERROR   1               ;FEC NOT EQUAL TO 14
007640  022767  007600  171622          CMP     #7600,FEA       ;CHECK FLOATING EXCEPTION ADDRESS
007646  001401                          BEQ     .+4             ;BRANCH IF OK
007650  104001                          ERROR   1               ;FEA NOT EQUAL TO 7600

007652  174067  171312                  STD     AC0,$REG0       ;GET RESULT FOR TYPING
007656  173467  172002                  CMPD    D40,AC0         ;DID IT CHANGE AC0?
007662  170267  171576                  STFPS   FPS             ;GET STATUS
007666  170000                          CFCC                    ;GET CC
007670  001401                          BEQ     .+4             ;SKIP IF OK
007672  104010                          ERROR   10              ;RESULT IS WRONG

007674  170127  040200                  LDFPS   #40200          ;DOUBLE MODE
007700  172467  171716                  LDF     DMZERO,AC0      ;LOAD A -0
007704  170267  171554                  STFPS   FPS             ;STORE FLOATING POINT STATUS
007710  022767  040214  171546          CMP     #40214,FPS      ;CHECK FLOATING POINT STATUS
007716  001401                          BEQ     .+4             ;BRANCH IF OK
007720  104001                          ERROR   1               ;FPS NOT EQUAL TO 40214

007722  174067  171242                  STD     AC0,$REG0       ;GET RESULT
007726  173467  171670                  CMPD    DMZERO,AC0      ;CMP TO -0
007732  170267  171526                  STFPS   FPS             ;GET STATUS
007736  170000                          CFCC                    ;GET CC
007740  001401                          BEQ     .+4             ;SKIP IF OK
007742  104010                          ERROR   10              ;RESULT NOT -0

                                ;;*******************************************************************
                                ;*TEST 27        MULF ERROR - OVERFLOW
                                ;;*******************************************************************
007744  000004                  TST27:  SCOPE
007746  170127  001000                  LDFPS   #1000           ;FLOATING/OVERFLOW
007752  005001                          CLR     R1              ;CLEAR FLAG WORD
007754  172427  076101                  LDF     #1E36,AC0       ;LOAD A LARGE NUMBER INTO AC0
007760  171027  076101          1$:     MULF    #1E36,AC0       ;MULTIPLY BY A LARGE NUMBER
007764  174067  171200                  STF     AC0,$REG0       ;GET FOR TYPING
007770  170267  171470                  STFPS   FPS             ;GET STATUS
007774  005701                          TST     R1              ;DID IT TRAP?
007776  001001                          BNE     3$              ;YES
010000  104007                          ERROR   7               ;DID NOT TRAP ON OVERFLOW
010002                          3$:
```

```
2372  010002  170267  171456                  STFPS   FPS             ;STORE FLOATING POINT STATUS
2373  010006  170367  171454                  STST    FEC             ;STORE EXCEPTION CODES
2374  010012  022767  101002  171444          CMP     #101002,FPS     ;CHECK FLOATING POINT STATUS
2375  010020  001401                          BEQ     .+4             ;BRANCH IF OK
2376  010022  104001                          ERROR   1               ;FPS NOT EQUAL TO 101002
2377  010024  022767  000010  171434          CMP     #10,FEC         ;CHECK FLOATING EXCEPTION CODE
2378  010032  001401                          BEQ     .+4             ;BRANCH IF OK
2379  010034  104001                          ERROR   1               ;FEC NOT EQUAL TO 10
2380  010036  022767  007760  171424          CMP     #7760,FEA       ;CHECK FLOATING EXCEPTION ADDRESS
2381  010044  001401                          BEQ     .+4             ;BRANCH IF OK
2382  010046  104001                          ERROR   1               ;FEA NOT EQUAL TO 7760
2383
2384                                  ;;*****************************************************************
2385                                  ;*TEST 30       DIVF ERROR - UNDERFLOW
2386                                  ;;*****************************************************************
2387  010050  000004                  TST30:  SCOPE
2388  010052  170127  002000                  LDFPS   #2000           ;FLOATING/UNDERFLOW
2389  010056  005001                          CLR     R1              ;CLEAR FLAG WORD
2390  010060  172427  002252                  LDF     #1E-36,AC0      ;LOAD A SMALL NUMBER
2391  010064  174427  076101          1$:     DIVF    #1E36,AC0       ;DIVIDE BY A LARGE NUMBER
2392  010070  174067  171074                  STF     AC0,$REG0       ;GET FOR TYPING
2393  010074  170267  171364                  STFPS   FPS             ;GET STATUS
2394  010100  005701                          TST     R1              ;DID IT TRAP?
2395  010102  001001                          BNE     3$              ;SKIP IF SET
2396  010104  104007                          ERROR   7               ;DID NOT TRAP ON UNDERFLOW
2397  010106                          3$:
2398  010106  170267  171352                  STFPS   FPS             ;STORE FLOATING POINT STATUS
2399  010112  170367  171350                  STST    FEC             ;STORE EXCEPTION CODES
2400  010116  022767  102000  171340          CMP     #102000,FPS     ;CHECK FLOATING POINT STATUS
2401  010124  001401                          BEQ     .+4             ;BRANCH IF OK
2402  010126  104001                          ERROR   1               ;FPS NOT EQUAL TO 102000
2403  010130  022767  000012  171330          CMP     #12,FEC         ;CHECK FLOATING EXCEPTION CODE
2404  010136  001401                          BEQ     .+4             ;BRANCH IF OK
2405  010140  104001                          ERROR   1               ;FEC NOT EQUAL TO 12
2406  010142  022767  010064  171320          CMP     #10064,FEA      ;CHECK FLOATING EXCEPTION ADDRESS
2407  010150  001401                          BEQ     .+4             ;BRANCH IF OK
2408  010152  104001                          ERROR   1               ;FEA NOT EQUAL TO 10064
2409
2410                                  ;;*****************************************************************
2411                                  ;*TEST 31       STCFI ERROR - CONVERSION(6)
2412                                  ;;*****************************************************************
2413  010154  000004                  TST31:  SCOPE
2414  010156  170127  000400                  LDFPS   #400            ;FLOATING/INTEGER/CONVERSION
2415  010162  005001                          CLR     R1              ;CLEAR FLAG WORD
2416  010164  172527  076101                  LDF     #1E36,AC1       ;LOAD LARGE NUMBER
2417  010170  175567  170774          1$:     STCFI   AC1,$REG0       ;TRY TO STUFF INTO 16 BITS
2418  010174  170267  171264                  STFPS   FPS             ;GET STATUS
2419  010200  005701                          TST     R1              ;TRAP FLAG SET?
2420  010202  001001                          BNE     3$              ;SKIP IF SET
2421  010204  104007                          ERROR   7               ;DID NOT TRAP ON CONVERT
2422  010206                          3$:
2423  010206  170267  171252                  STFPS   FPS             ;STORE FLOATING POINT STATUS
2424  010212  170367  171250                  STST    FEC             ;STORE EXCEPTION CODES
2425  010216  022767  100405  171240          CMP     #100405,FPS     ;CHECK FLOATING POINT STATUS
2426  010224  001401                          BEQ     .+4             ;BRANCH IF OK
2427  010226  104001                          ERROR   1               ;FPS NOT EQUAL TO 100405
```

```
2428  010230  022767  000006  171230          CMP     #6,FEC          ;CHECK FLOATING EXCEPTION CODE
2429  010236  001401                          BEQ     .+4             ;BRANCH IF OK
2430  010240  104001                          ERROR   1               ;FEC NOT EQUAL TO 6
2431  010242  022767  010170  171220          CMP     #10170,FEA      ;CHECK FLOATING EXCEPTION ADDRESS
2432  010250  001401                          BEQ     .+4             ;BRANCH IF OK
2433  010252  104001                          ERROR   1               ;FEA NOT EQUAL TO 10170
2434
2435                                  ;;*****************************************************************
2436                                  ;*TEST 32        DIVF BY 0 ERROR
2437                                  ;;*****************************************************************
2438  010254  000004                  TST32:  SCOPE
2439  010256  170127  000000                  LDFPS   #0              ;FLOATING
2440  010262  005001                          CLR     R1              ;CLEAR FLAG
2441  010264  174527  000000          1$:     DIVF    #0,AC1          ;DIVIDE BY 0
2442  010270  170267  171170                  STFPS   FPS             ;GET STATUS
2443  010274  005701                          TST     R1              ;CHECK FLAG
2444  010276  001001                          BNE     3$              ;SKIP IF SET
2445  010300  104007                          ERROR   7               ;DIVIDE BY 0 DID NOT TRAP
2446  010302                          3$:
2447  010302  170267  171156                  STFPS   FPS             ;STORE FLOATING POINT STATUS
2448  010306  170367  171154                  STST    FEC             ;STORE EXCEPTION CODES
2449  010312  022767  100000  171144          CMP     #100000,FPS     ;CHECK FLOATING POINT STATUS
2450  010320  001401                          BEQ     .+4             ;BRANCH IF OK
2451  010322  104001                          ERROR   1               ;FPS NOT EQUAL TO 100000
2452  010324  022767  000004  171134          CMP     #4,FEC          ;CHECK FLOATING EXCEPTION CODE
2453  010332  001401                          BEQ     .+4             ;BRANCH IF OK
2454  010334  104001                          ERROR   1               ;FEC NOT EQUAL TO 4
2455  010336  022767  010264  171124          CMP     #10264,FEA      ;CHECK FLOATING EXCEPTION ADDRESS
2456  010344  001401                          BEQ     .+4             ;BRANCH IF OK
2457  010346  104001                          ERROR   1               ;FEA NOT EQUAL TO 10264
2458
2459                                  ;;*****************************************************************
2460                                  ;*TEST 33        LDF -0 ERROR
2461                                  ;;*****************************************************************
2462  010350  000004                  TST33:  SCOPE
2463  010352  170127  004000                  LDFPS   #4000           ;FLOATING/UNDEFINED VARIABLE
2464  010356  005001                          CLR     R1              ;CLEAR FLAG
2465  010360  172667  171236          1$:     LDF     DMZERO,AC2      ;LOAD AN UNDEFINED VARIABLE
2466  010364  170267  171074                  STFPS   FPS             ;GET STATUS
2467  010370  005701                          TST     R1              ;CHECK FLAG
2468  010372  001001                          BNE     3$              ;SKIP IF SET
2469  010374  104007                          ERROR   7               ;LOAD OF -0 DID NOT TRAP
2470  010376                          3$:
2471  010376  170267  171062                  STFPS   FPS             ;STORE FLOATING POINT STATUS
2472  010402  170367  171060                  STST    FEC             ;STORE EXCEPTION CODES
2473  010406  022767  104014  171050          CMP     #104014,FPS     ;CHECK FLOATING POINT STATUS
2474  010414  001401                          BEQ     .+4             ;BRANCH IF OK
2475  010416  104001                          ERROR   1               ;FPS NOT EQUAL TO 104014
2476  010420  022767  000014  171040          CMP     #14,FEC         ;CHECK FLOATING EXCEPTION CODE
2477  010426  001401                          BEQ     .+4             ;BRANCH IF OK
2478  010430  104001                          ERROR   1               ;FEC NOT EQUAL TO 14
2479  010432  022767  010360  171030          CMP     #10360,FEA      ;CHECK FLOATING EXCEPTION ADDRESS
2480  010440  001401                          BEQ     .+4             ;BRANCH IF OK
2481  010442  104001                          ERROR   1               ;FEA NOT EQUAL TO 10360
2482
2483                                  ;;*****************************************************************
```

```
2484                                        ;*TEST 34       OPCODE ERROR
2485                                        ;;*****************************************************************
2486  010444  000004                  TST34:  SCOPE
2487  010446  170127  000000                  LDFPS   #0              ;FLOATING
2488  010452  005001                          CLR     R1              ;CLEAR FLAG
2489  010454  177707                  1$:     177707                  ;ILLEGAL OPCOPE
2490  010456  170267  171002                  STFPS   FPS             ;GET STATUS
2491  010462  005701                          TST     R1              ;CHECK FLAG
2492  010464  001001                          BNE     3$              ;SKIP IF SET
2493  010466  104007                          ERROR   7               ;NOT AN ILLEGAL OPCODE
2494  010470                          3$:
2495  010470  170267  170770                  STFPS   FPS             ;STORE FLOATING POINT STATUS
2496  010474  170367  170766                  STST    FEC             ;STORE EXCEPTION CODES
2497  010500  022767  100000  170756          CMP     #100000,FPS     ;CHECK FLOATING POINT STATUS
2498  010506  001401                          BEQ     .+4             ;BRANCH IF OK
2499  010510  104001                          ERROR   1               ;FPS NOT EQUAL TO 100000
2500  010512  022767  000002  170746          CMP     #2,FEC          ;CHECK FLOATING EXCEPTION CODE
2501  010520  001401                          BEQ     .+4             ;BRANCH IF OK
2502  010522  104001                          ERROR   1               ;FEC NOT EQUAL TO 2
2503  010524  022767  010454  170736          CMP     #10454,FEA      ;CHECK FLOATING EXCEPTION ADDRESS
2504  010532  001401                          BEQ     .+4             ;BRANCH IF OK
2505  010534  104001                          ERROR   1               ;FEA NOT EQUAL TO 10454
2506
2507                                        ;;*****************************************************************
2508                                        ;*TEST 35       ADDF ERROR - OVERFLOW
2509                                        ;;*****************************************************************
2510  010536  000004                  TST35:  SCOPE
2511  010540  170127  001000                  LDFPS   #1000           ;FLOATING/OVERFLOW
2512  010544  005001                          CLR     R1              ;CLEAR FLAG
2513  010546  172767  171040                  LDF     DBIG,AC3        ;LOAD A BIG NUMBER
2514  010552  172367  171034          1$:     ADDF    DBIG,AC3        ;MAKE OVERFLOW
2515  010556  170267  170702                  STFPS   FPS             ;GET STATUS
2516  010562  174367  170402                  STF     AC3,$REG0       ;GET RESULT
2517  010566  005701                          TST     R1              ;FLAG SET?
2518  010570  001001                          BNE     3$              ;SKIP IF SET
2519  010572  104007                          ERROR   7               ;DID NOT TRAP ON OVERFLOW
2520  010574                          3$:
2521  010574  170267  170664                  STFPS   FPS             ;STORE FLOATING POINT STATUS
2522  010600  170367  170662                  STST    FEC             ;STORE EXCEPTION CODES
2523  010604  022767  101006  170652          CMP     #101006,FPS     ;CHECK FLOATING POINT STATUS
2524  010612  001401                          BEQ     .+4             ;BRANCH IF OK
2525  010614  104001                          ERROR   1               ;FPS NOT EQUAL TO 101006
2526  010616  022767  000010  170642          CMP     #10,FEC         ;CHECK FLOATING EXCEPTION CODE
2527  010624  001401                          BEQ     .+4             ;BRANCH IF OK
2528  010626  104001                          ERROR   1               ;FEC NOT EQUAL TO 10
2529  010630  022767  010552  170632          CMP     #10552,FEA      ;CHECK FLOATING EXCEPTION ADDRESS
2530  010636  001401                          BEQ     .+4             ;BRANCH IF OK
2531  010640  104001                          ERROR   1               ;FEA NOT EQUAL TO 10552
2532
2533                                        ;;*****************************************************************
2534                                        ;*TEST 36       SUBF ERROR - UNDERFLOW
2535                                        ;;*****************************************************************
2536  010642  000004                  TST36:  SCOPE
2537  010644  170127  002000                  LDFPS   #2000           ;FLOATING/UNDERFLOW
2538  010650  005001                          CLR     R1              ;CLEAR FLAG
2539  010652  172427  000430                  LDF     #.07E-37,AC0    ;LOAD SMALL NUMBER
```

```
2540  010656  173027  000504          1$:   SUBF    #.09E-37,AC0    ;SUBF SMALL NUMBER
2541  010662  174067  170302                STF     AC0,$REG0       ;GET RESULT
2542  010666  170267  170572                STFPS   FPS             ;GET STATUS
2543  010672  005701                       TST     R1              ;FLAG SET?
2544  010674  001001                       BNE     3$              ;SKIP IF SET
2545  010676  104007                       ERROR   7               ;NO TRAP ON UNDERFLOW
2546  010700                        3$:
2547  010700  170267  170560                STFPS   FPS             ;STORE FLOATING POINT STATUS
2548  010704  170367  170556                STST    FEC             ;STORE EXCEPTION CODES
2549  010710  022767  102014  170546        CMP     #102014,FPS     ;CHECK FLOATING POINT STATUS
2550  010716  001401                       BEQ     .+4             ;BRANCH IF OK
2551  010720  104001                       ERROR   1               ;FPS NOT EQUAL TO 102014
2552  010722  022767  000012  170536        CMP     #12,FEC         ;CHECK FLOATING EXCEPTION CODE
2553  010730  001401                       BEQ     .+4             ;BRANCH IF OK
2554  010732  104001                       ERROR   1               ;FEC NOT EQUAL TO 12
2555  010734  022767  010656  170526        CMP     #10656,FEA      ;CHECK FLOATING EXCEPTION ADDRESS
2556  010742  001401                       BEQ     .+4             ;BRANCH IF OK
2557  010744  104001                       ERROR   1               ;FEA NOT EQUAL TO 10656
2558
2559
2560                                  ;;*****************************************************************
2561                                  ;*TEST 37       TEST OF FEC-02, WITH NO TRAP, INTERRUPT DISABLED
2562                                  ;;*****************************************************************
2563  010746  000004                TST37:  SCOPE
2564  010750  012704  014300                MOV     #FEC02,R4       ; PTR TO FPP FEC CODE GENERATOR
2565  010754  012705  011002                MOV     #FPPN1,R5       ; PTR TO TEST DATA SET
2566  010760  012767  014300  170512        MOV     #FII02,EXPFEA   ; ADDR(FPP BAD INSTR) EXPECTED
2567  010766  012767  014302  170506        MOV     #NXT02,EXPRTS   ; ADDR(NEXT INSTR) EXPECTED
2568
2569  010774  004737  013744                JSR     PC,@#TRPTST     ; GO TEST
2570
2571  011000                        64$:
2572  011000  000406                       BR      TST40           ;;
2573
2574  011002                        FPPN1:  ; TEST DATA SET FPPN-1:
2575  011002  000157  000040  000157        .WORD   000157,000040,000157    ; PSW'S:  BEFORE, LOADED, AFTER
2576  011010  047417  147417  100002        .WORD   047417,147417,100002    ; OLDFPS/NEWFPS/FECCODE
2577
2578
2579                                  ;;*****************************************************************
2580                                  ;*TEST 40       TEST OF FEC-04, WITH NO TRAP, INTERRUPT DISABLED
2581                                  ;;*****************************************************************
2582  011016  000004                TST40:  SCOPE
2583  011020  012704  014310                MOV     #FEC04,R4       ; PTR TO FPP FEC CODE GENERATOR
2584  011024  012705  011052                MOV     #FPPN2,R5       ; PTR TO TEST DATA SET
2585  011030  012767  014314  170442        MOV     #FII04,EXPFEA   ; ADDR(FPP BAD INSTR) EXPECTED
2586  011036  012767  014320  170436        MOV     #NXT04,EXPRTS   ; ADDR(NEXT INSTR) EXPECTED
2587
2588  011044  004737  013744                JSR     PC,@#TRPTST     ; GO TEST
2589
2590  011050                        64$:
2591  011050  000406                       BR      TST41           ;;
2592
2593  011052                        FPPN2:  ; TEST DATA SET FPPN-2:
2594  011052  000157  000040  000157        .WORD   000157,000040,000157    ; PSW'S:  BEFORE, LOADED, AFTER
2595  011060  047417  147400  100004        .WORD   047417,147400,100004    ; OLDFPS/NEWFPS/FECCODE
```

```
2596
2597
2598                                    ;;*****************************************************************
2599                                    ;*TEST 41       TEST OF FEC-06, WITH NO TRAP, INTERRUPT DISABLED
2600                                    ;;*****************************************************************
2601  011066  000004                    TST41:  SCOPE
2602  011070  012704  014326                    MOV     #FEC06,R4       ; PTR TO FPP FEC CODE GENERATOR
2603  011074  012705  011122                    MOV     #FPPN3,R5       ; PTR TO TEST DATA SET
2604  011100  012767  014332  170372            MOV     #FII06,EXPFEA   ; ADDR(FPP BAD INSTR) EXPECTED
2605  011106  012767  014336  170366            MOV     #NXT06,EXPRTS   ; ADDR(NEXT INSTR) EXPECTED
2606
2607  011114  004737  013744                    JSR     PC,@#TRPTST     ; GO TEST
2608
2609  011120                            64$:
2610  011120  000406                            BR      TST42           ;;
2611
2612  011122                            FPPN3:  ; TEST DATA SET FPPN-3:
2613  011122  000155  000042  000142            .WORD   000155,000042,000142    ; PSW'S:  BEFORE, LOADED, AFTER
2614  011130  047412  147405  100006            .WORD   047412,147405,100006    ; OLDFPS/NEWFPS/FECCODE
2615
2616
2617                                    ;;*****************************************************************
2618                                    ;*TEST 42       TEST OF FEC-10, WITH NO TRAP, INTERRUPT DISABLED
2619                                    ;;*****************************************************************
2620  011136  000004                    TST42:  SCOPE
2621  011140  012704  014344                    MOV     #FEC10,R4       ; PTR TO FPP FEC CODE GENERATOR
2622  011144  012705  011172                    MOV     #FPPN4,R5       ; PTR TO TEST DATA SET
2623  011150  012767  014354  170322            MOV     #FII10,EXPFEA   ; ADDR(FPP BAD INSTR) EXPECTED
2624  011156  012767  014356  170316            MOV     #NXT10,EXPRTS   ; ADDR(NEXT INSTR) EXPECTED
2625
2626  011164  004737  013744                    JSR     PC,@#TRPTST     ; GO TEST
2627
2628  011170                            64$:
2629  011170  000406                            BR      TST43           ;;
2630
2631  011172                            FPPN4:  ; TEST DATA SET FPPN-4:
2632  011172  000151  000046  000151            .WORD   000151,000046,000151    ; PSW'S:  BEFORE, LOADED, AFTER
2633  011200  047411  147406  100010            .WORD   047411,147406,100010    ; OLDFPS/NEWFPS/FECCODE
2634
2635
2636                                    ;;*****************************************************************
2637                                    ;*TEST 43       TEST OF FEC-12, WITH NO TRAP, INTERRUPT DISABLED
2638                                    ;;*****************************************************************
2639  011206  000004                    TST43:  SCOPE
2640  011210  012704  014364                    MOV     #FEC12,R4       ; PTR TO FPP FEC CODE GENERATOR
2641  011214  012705  011242                    MOV     #FPPN5,R5       ; PTR TO TEST DATA SET
2642  011220  012767  014370  170252            MOV     #FII12,EXPFEA   ; ADDR(FPP BAD INSTR) EXPECTED
2643  011226  012767  014374  170246            MOV     #NXT12,EXPRTS   ; ADDR(NEXT INSTR) EXPECTED
2644
2645  011234  004737  013744                    JSR     PC,@#TRPTST     ; GO TEST
2646
2647  011240                            64$:
2648  011240  000406                            BR      TST44           ;;
2649
2650  011242                            FPPN5:  ; TEST DATA SET FPPN-5:
2651  011242  000157  000040  000157            .WORD   000157,000040,000157    ; PSW'S:  BEFORE, LOADED, AFTER
```

```
FPU INSTR EXERCISER     MACY11 27(1006)  09-FEB-77  10:28  PAGE 63
DQFPCA.P11    09-FEB-77 10:26            T43     TEST OF FEC-12, WITH NO TRAP, INTERRUPT DISABLED

  2652  011250  047417  147400  100012          .WORD   047417,147400,100012    ; OLDFPS/NEWFPS/FECCODE
  2653
  2654
  2655                                  ;;*****************************************************************
  2656                                  ;*TEST 44       TEST OF FEC-14, WITH NO TRAP, INTERRUPT DISABLED
  2657                                  ;;*****************************************************************
  2658  011256  000004                 TST44:  SCOPE
  2659  011260  012704  014402                 MOV     #FEC14,R4       ; PTR TO FPP FEC CODE GENERATOR
  2660  011264  012705  011312                 MOV     #FPPN6,R5       ; PTR TO TEST DATA SET
  2661  011270  012767  014402  170202         MOV     #FII14,EXPFEA   ; ADDR(FPP BAD INSTR) EXPECTED
  2662  011276  012767  014406  170176         MOV     #NXT14,EXPRTS   ; ADDR(NEXT INSTR) EXPECTED
  2663
  2664  011304  004737  013744                 JSR     PC,@#TRPTST     ; GO TEST
  2665
  2666  011310                          64$:
  2667  011310  000406                         BR      TST45           ;;
  2668
  2669  011312                          FPPN6:  ; TEST DATA SET FPPN-6:
  2670  011312  000143  000054  000143         .WORD   000143,000054,000143    ; PSW'S:  BEFORE, LOADED, AFTER
  2671  011320  047403  147414  100014         .WORD   047403,147414,100014    ; OLDFPS/NEWFPS/FECCODE
  2672
  2673
  2674                                  ;;*****************************************************************
  2675                                  ;*TEST 45       TEST OF FEC-16, WITH NO TRAP, INTERRUPT DISABLED
  2676                                  ;;*****************************************************************
  2677  011326  000004                 TST45:  SCOPE
  2678
  2679                                  ;/////////////////////////////////////////////////////////////////
  2680                                          ;THIS TEST IS EXECUTED FOR HFP ONLY, NOT IN WFP MODE
  2681
  2682  011330  076600  000022                 MFD     #WHAMI          ;GET WHAMI INTO R0
  2683  011334  032700  000020                 BIT     #BIT04,R0       ;HFP IN SYSTEM ?
  2684  011340  001421                         BEQ     64$             ;BR IF NONE
  2685
  2686  011342  076600  000144                 MFD     #FLAG           ;GET FLAGS INTO R0
  2687  011346  032700  010000                 BIT     #BIT12,R0       ;HFP ENABLED ?
  2688  011352  001414                         BEQ     64$             ;BR IF NOT
  2689
  2690                                  ;/////////////////////////////////////////////////////////////////
  2691
  2692  011354  012704  014414                 MOV     #FEC16,R4       ; PTR TO FPP FEC CODE GENERATOR
  2693  011360  012705  011406                 MOV     #FPPN7,R5       ; PTR TO TEST DATA SET
  2694  011364  012767  014422  170106         MOV     #FII16,EXPFEA   ; ADDR(FPP BAD INSTR) EXPECTED
  2695  011372  012767  014424  170102         MOV     #NXT16,EXPRTS   ; ADDR(NEXT INSTR) EXPECTED
  2696
  2697  011400  004737  013744                 JSR     PC,@#TRPTST     ; GO TEST
  2698
  2699  011404                          64$:
  2700  011404  000406                         BR      TST46           ;;
  2701
  2702  011406                          FPPN7:  ; TEST DATA SET FPPN-7:
  2703  011406  000157  000052  000140         .WORD   000157,000052,000140    ; PSW'S:  BEFORE, LOADED, AFTER
  2704  011414  047420  147420  100016         .WORD   047420,147420,100016    ; OLDFPS/NEWFPS/FECCODE
  2705
  2706
  2707
```

```
                                        ;////////////////////////////////////////////////////////////////////////
                                        ;
                                        ;       NOTE:  IN THE FOLLOWING GROUP OF TESTS, THE LOADED PS
                                        ;              MUST CONTAIN PROCESSOR PRIORITY OF (6) OR GREATER
                                        ;              TO LOCK OUT ANY CLOCK INTERRUPTS (AT BR6).   THEY
                                        ;              ARE SETUP HERE WITH PRIORITY (7).
                                        ;
                                        ;////////////////////////////////////////////////////////////////////////

                                        ;;*********************************************************************
                                        ;*TEST 46       TEST OF FEC-02, WITH TRAP, INTERRUPT ENABLED
                                        ;;*********************************************************************
011422  000004                          TST46:  SCOPE
011424  012704  014300                          MOV     #FEC02,R4       ; PTR TO FPP FEC CODE GENERATOR
011430  012705  011456                          MOV     #FPPT10,R5      ; PTR TO TEST DATA SET
011434  012767  014300  170036                  MOV     #FII02,EXPFEA   ; ADDR(FPP BAD INSTR) EXPECTED
011442  012767  014302  170032                  MOV     #NXT02,EXPRTS   ; ADDR(NEXT INSTR) EXPECTED

011450  004737  013744                          JSR     PC,@#TRPTST     ; GO TEST

011454                                  64$:
011454  000406                          TST47   BR      TST47           ;;

011456                                  FPPT10: ; TEST DATA SET FPPT-10:
011456  000353  000104  000353                  .WORD   000353,000104,000353    ; PSW'S:  BEFORE, LOADED, AFTER
011464  000017  100017  140002                  .WORD   000017,100017,140002    ; OLDFPS/NEWFPS/FECCODE


                                        ;;*********************************************************************
                                        ;*TEST 47       TEST OF FEC-04, WITH TRAP, INTERRUPT ENABLED
                                        ;;*********************************************************************
011472  000004                          TST47:  SCOPE
011474  012704  014310                          MOV     #FEC04,R4       ; PTR TO FPP FEC CODE GENERATOR
011500  012705  011526                          MOV     #FPPT11,R5      ; PTR TO TEST DATA SET
011504  012767  014314  167766                  MOV     #FII04,EXPFEA   ; ADDR(FPP BAD INSTR) EXPECTED
011512  012767  014320  167762                  MOV     #NXT04,EXPRTS   ; ADDR(NEXT INSTR) EXPECTED

011520  004737  013744                          JSR     PC,@#TRPTST     ; GO TEST

011524                                  64$:
011524  000406                          TST50   BR      TST50           ;;

011526                                  FPPT11: ; TEST DATA SET FPPT-11:
011526  000350  000107  000350                  .WORD   000350,000107,000350    ; PSW'S:  BEFORE, LOADED, AFTER
011534  000017  100000  140004                  .WORD   000017,100000,140004    ; OLDFPS/NEWFPS/FECCODE


                                        ;;*********************************************************************
                                        ;*TEST 50       TEST OF FEC-06, WITH NO TRAP, INTERRUPT ENABLED
                                        ;;*********************************************************************
011542  000004                          TST50:  SCOPE
011544  012704  014326                          MOV     #FEC06,R4       ; PTR TO FPP FEC CODE GENERATOR
011550  012705  011576                          MOV     #FPPN12,R5      ; PTR TO TEST DATA SET
011554  012767  014332  167716                  MOV     #FII06,EXPFEA   ; ADDR(FPP BAD INSTR) EXPECTED
011562  012767  014336  167712                  MOV     #NXT06,EXPRTS   ; ADDR(NEXT INSTR) EXPECTED
```

```
                                        T50     TEST OF FEC-06, WITH NO TRAP, INTERRUPT ENABLED

2764  011570  004737  013744                            JSR     PC,@#TRPTST     ; GO TEST
2765
2766  011574                          64$:
2767  011574  000406                            BR      TST51           ;;
2768
2769  011576                          FPPN12: ; TEST DATA SET FPPN-12:
2770  011576  000353  000104  000345           .WORD   000353,000104,000345    ; PSW'S:  BEFORE, LOADED, AFTER
2771  011604  007012  007005  000000           .WORD   007012,007005,000000    ; OLDFPS/NEWFPS/FECCODE
2772
2773
2774                                    ;;******************************************************************
2775                                    ;*TEST 51       TEST OF FEC-06, WITH TRAP, INTERRUPT ENABLED
2776                                    ;;******************************************************************
2777  011612  000004                    TST51:  SCOPE
2778  011614  012704  014326                    MOV     #FEC06,R4       ; PTR TO FPP FEC CODE GENERATOR
2779  011620  012705  011646                    MOV     #FPPT13,R5      ; PTR TO TEST DATA SET
2780  011624  012767  014332  167646            MOV     #FII06,EXPFEA   ; ADDR(FPP BAD INSTR) EXPECTED
2781  011632  012767  014336  167642            MOV     #NXT06,EXPRTS   ; ADDR(NEXT INSTR) EXPECTED
2782
2783  011640  004737  013744                    JSR     PC,@#TRPTST     ; GO TEST
2784
2785  011644                          64$:
2786  011644  000406                            BR      TST52           ;;
2787
2788  011646                          FPPT13: ; TEST DATA SET FPPT-13:
2789  011646  000353  000104  000345           .WORD   000353,000104,000345    ; PSW'S:  BEFORE, LOADED, AFTER
2790  011654  000412  100405  140006           .WORD   000412,100405,140006    ; OLDFPS/NEWFPS/FECCODE
2791
2792
2793                                    ;;******************************************************************
2794                                    ;*TEST 52       TEST OF FEC-10, WITH NO TRAP, INTERRUPT ENABLED
2795                                    ;;******************************************************************
2796  011662  000004                    TST52:  SCOPE
2797  011664  012704  014344                    MOV     #FEC10,R4       ; PTR TO FPP FEC CODE GENERATOR
2798  011670  012705  011716                    MOV     #FPPN14,R5      ; PTR TO TEST DATA SET
2799  011674  012767  014354  167576            MOV     #FII10,EXPFEA   ; ADDR(FPP BAD INSTR) EXPECTED
2800  011702  012767  014356  167572            MOV     #NXT10,EXPRTS   ; ADDR(NEXT INSTR) EXPECTED
2801
2802  011710  004737  013744                    JSR     PC,@#TRPTST     ; GO TEST
2803
2804  011714                          64$:
2805  011714  000406                            BR      TST53           ;;
2806
2807  011716                          FPPN14: ; TEST DATA SET FPPN-14:
2808  011716  000350  000107  000350           .WORD   000350,000107,000350    ; PSW'S:  BEFORE, LOADED, AFTER
2809  011724  006411  006406  000000           .WORD   006411,006406,000000    ; OLDFPS/NEWFPS/FECCODE
2810
2811
2812                                    ;;******************************************************************
2813                                    ;*TEST 53       TEST OF FEC-10, WITH TRAP, INTERRUPT ENABLED
2814                                    ;;******************************************************************
2815  011732  000004                    TST53:  SCOPE
2816  011734  012704  014344                    MOV     #FEC10,R4       ; PTR TO FPP FEC CODE GENERATOR
2817  011740  012705  011766                    MOV     #FPPT15,R5      ; PTR TO TEST DATA SET
2818  011744  012767  014354  167526            MOV     #FII10,EXPFEA   ; ADDR(FPP BAD INSTR) EXPECTED
2819  011752  012767  014356  167522            MOV     #NXT10,EXPRTS   ; ADDR(NEXT INSTR) EXPECTED
```

```
                                
012760  004737  013744                JSR     PC,@#TRPTST     ; GO TEST
                                
011764                          64$:
011764  000406                  BR      TST54           ;;
                                
011766                          FPPT15: ; TEST DATA SET FPPT-15:
011766  000350  000107  000350          .WORD   000350,000107,000350    ; PSW'S:  BEFORE, LOADED, AFTER
011774  001011  101006  140010          .WORD   001011,101006,140010    ; OLDFPS/NEWFPS/FECCODE
                                
                                
                                ;;*****************************************************************
                                ;*TEST 54       TEST OF FEC-12, WITH NO TRAP, INTERRUPT ENABLED
                                ;;*****************************************************************
012002  000004                  TST54:  SCOPE
012004  012704  014364                  MOV     #FEC12,R4       ; PTR TO FPP FEC CODE GENERATOR
012010  012705  012036                  MOV     #FPPN16,R5      ; PTR TO TEST DATA SET
012014  012767  014370  167456          MOV     #FII12,EXPFEA   ; ADDR(FPP BAD INSTR) EXPECTED
012022  012767  014374  167452          MOV     #NXT12,EXPRTS   ; ADDR(NEXT INSTR) EXPECTED
                                
012030  004737  013744                JSR     PC,@#TRPTST     ; GO TEST
                                
012034                          64$:
012034  000406                  BR      TST55           ;;
                                
012036                          FPPN16: ; TEST DATA SET FPPN-16:
012036  000350  000107  000350          .WORD   000350,000107,000350    ; PSW'S:  BEFORE, LOADED, AFTER
012044  005413  005404  000000          .WORD   005413,005404,000000    ; OLDFPS/NEWFPS/FECCODE
                                
                                
                                ;;*****************************************************************
                                ;*TEST 55       TEST OF FEC-12, WITH TRAP, INTERRUPT ENABLED
                                ;;*****************************************************************
012052  000004                  TST55:  SCOPE
012054  012704  014364                  MOV     #FEC12,R4       ; PTR TO FPP FEC CODE GENERATOR
012060  012705  012106                  MOV     #FPPT17,R5      ; PTR TO TEST DATA SET
012064  012767  014370  167406          MOV     #FII12,EXPFEA   ; ADDR(FPP BAD INSTR) EXPECTED
012072  012767  014374  167402          MOV     #NXT12,EXPRTS   ; ADDR(NEXT INSTR) EXPECTED
                                
012100  004737  013744                JSR     PC,@#TRPTST     ; GO TEST
                                
012104                          64$:
012104  000406                  BR      TST56           ;;
                                
012106                          FPPT17: ; TEST DATA SET FPPT-17:
012106  000350  000107  000350          .WORD   000350,000107,000350    ; PSW'S:  BEFORE, LOADED, AFTER
012114  002017  102000  140012          .WORD   002017,102000,140012    ; OLDFPS/NEWFPS/FECCODE
                                
                                
                                ;;*****************************************************************
                                ;*TEST 56       TEST OF FEC-14, WITH NO TRAP, INTERRUPT ENABLED
                                ;;*****************************************************************
012122  000004                  TST56:  SCOPE
012124  012704  014402                  MOV     #FEC14,R4       ; PTR TO FPP FEC CODE GENERATOR
012130  012705  012156                  MOV     #FPPN20,R5      ; PTR TO TEST DATA SET
012134  012767  014402  167336          MOV     #FII14,EXPFEA   ; ADDR(FPP BAD INSTR) EXPECTED
```

```
                                T56     TEST OF FEC-14, WITH NO TRAP, INTERRUPT ENABLED

2876  012142  012767  014406  167332            MOV     #NXT14,EXPRTS   ; ADDR(NEXT INSTR) EXPECTED
2877
2878  012150  004737  013744                    JSR     PC,@#TRPTST     ; GO TEST
2879
2880  012154                          64$:
2881  012154  000406                            BR      TST57           ;;
2882
2883  012156                          FPPN20: ; TEST DATA SET FPPN-20:
2884  012156  000343  000114  000343            .WORD   000343,000114,000343    ; PSW'S:  BEFORE, LOADED, AFTER
2885  012164  003403  003414  000000            .WORD   003403,003414,000000    ; OLDFPS/NEWFPS/FECCODE
2886
2887
2888                                ;;**********************************************************************
2889                                ;*TEST 57       TEST OF FEC-14, WITH TRAP, INTERRUPT ENABLED
2890                                ;;**********************************************************************
2891  012172  000004                TST57:  SCOPE
2892  012174  012704  014402                    MOV     #FEC14,R4       ; PTR TO FPP FEC CODE GENERATOR
2893  012200  012705  012226                    MOV     #FPPT21,R5      ; PTR TO TEST DATA SET
2894  012204  012767  014402  167266            MOV     #FII14,EXPFEA   ; ADDR(FPP BAD INSTR) EXPECTED
2895  012212  012767  014406  167262            MOV     #NXT14,EXPRTS   ; ADDR(NEXT INSTR) EXPECTED
2896
2897  012220  004737  013744                    JSR     PC,@#TRPTST     ; GO TEST
2898
2899  012224                          64$:
2900  012224  000406                            BR      TST60           ;;
2901
2902  012226                          FPPT21: ; TEST DATA SET FPPT-21:
2903  012226  000343  000114  000343            .WORD   000343,000114,000343    ; PSW'S:  BEFORE, LOADED, AFTER
2904  012234  004003  104014  140014            .WORD   004003,104014,140014    ; OLDFPS/NEWFPS/FECCODE
2905
2906
2907                                ;;**********************************************************************
2908                                ;*TEST 60       TEST OF FEC-16, WITH NO TRAP, INTERRUPT ENABLED
2909                                ;;**********************************************************************
2910  012242  000004                TST60:  SCOPE
2911  012244  012704  014414                    MOV     #FEC16,R4       ; PTR TO FPP FEC CODE GENERATOR
2912  012250  012705  012276                    MOV     #FPPN22,R5      ; PTR TO TEST DATA SET
2913  012254  012767  014422  167216            MOV     #FII16,EXPFEA   ; ADDR(FPP BAD INSTR) EXPECTED
2914  012262  012767  014424  167212            MOV     #NXT16,EXPRTS   ; ADDR(NEXT INSTR) EXPECTED
2915
2916  012270  004737  013744                    JSR     PC,@#TRPTST     ; GO TEST
2917
2918  012274                          64$:
2919  012274  000406                            BR      TST61           ;;
2920
2921  012276                          FPPN22: ; TEST DATA SET FPPN-22:
2922  012276  000357  000145  000340            .WORD   000357,000145,000340    ; PSW'S:  BEFORE, LOADED, AFTER
2923  012304  007400  007400  000000            .WORD   007400,007400,000000    ; OLDFPS/NEWFPS/FECCODE
2924
2925
2926                                ;;**********************************************************************
2927                                ;*TEST 61       TEST OF FEC-16, WITH TRAP, INTERRUPT ENABLED
2928                                ;;**********************************************************************
2929  012312  000004                TST61:  SCOPE
2930
2931                                ;//////////////////////////////////////////////////////////////////////
```

```
2932                                                    ;THIS TEST IS EXECUTED FOR HFP ONLY, NOT IN WFP MODE
2933
2934  012314  076600  000022                            MFD     #RWHAMI         ;GET WHAMI INTO R0
2935  012320  032700  000020                            BIT     #BIT04,R0       ;HFP IN SYSTEM ?
2936  012324  001421                                    BEQ     64$             ;BR IF NONE
2937
2938  012326  076600  000144                            MFD     #RFLAG          ;GET FLAGS INTO R0
2939  012332  032700  010000                            BIT     #BIT12,R0       ;HFP ENABLED ?
2940  012336  001414                                    BEQ     64$             ;BR IF NOT
2941
2942                                                  ;//////////////////////////////////////////////////////////////////////
2943
2944  012340  012704  014414                            MOV     #FEC16,R4       ; PTR TO FPP FEC CODE GENERATOR
2945  012344  012705  012372                            MOV     #FPPT23,R5      ; PTR TO TEST DATA SET
2946  012350  012767  014422  167122                    MOV     #FII16,EXPFEA   ; ADDR(FPP BAD INSTR) EXPECTED
2947  012356  012767  014424  167116                    MOV     #NXT16,EXPRTS   ; ADDR(NEXT INSTR) EXPECTED
2948
2949  012364  004737  013744                            JSR     PC,@#TRPTST     ; GO TEST
2950
2951  012370                                          64$:
2952  012370  000406                                    BR      TST62           ;;
2953
2954  012372                                          FPPT23: ; TEST DATA SET FPPT-23:
2955  012372  000357  000152  000341                    .WORD   000357,000152,000341    ; PSW'S:  BEFORE, LOADED, AFTER
2956  012400  000020  100020  140016                    .WORD   000020,100020,140016    ; OLDFPS/NEWFPS/FECCODE
2957
2958
```

```
                                ;;**********************************************************************
                                ;*TEST 62       TEST FOR CONVERSION, ADDD AND SUBD
                                ;;**********************************************************************
012406  000004                  TST62:  SCOPE
012410  170127  047600                  LDFPS   #47600
012414  177327  000004                  LDCID   #4,AC3          ;LOAD AC3 WITH A FLOATING 4
012420  177227  000002                  LDCID   #2,AC2          ;LOAD AC2 WITH A FLOATING 2
012424  172302                          ADDD    AC2,AC3         ;ADD 2+4 = 6 IN AC3
012426  173302                          SUBD    AC2,AC3         ;SUB 2 FROM 6
012430  175737  001170                  STCDI   AC3,@#$REGO     ;@#$REGO SHOULD =4
012434  022737  000004  001170          CMP     #4,@#$REGO      ;DOES @#$REGO =4
012442  170267  167016                  STFPS   FPS             ;GET STATUS
012446  001401                          BEQ     .+4             ;YES
012450  104012                          ERROR   12              ;@#$REGO SHOULD = 4

                                ;;**********************************************************************
                                ;*TEST 63       LDD AND STD TEST
                                ;;**********************************************************************
012452  000004                  TST63:  SCOPE
012454  177027  000005                  LDCID   #5,AC0          ;LOAD AC0 WITH A FLOATING 5
012460  174005                          STD     AC0,AC5         ;NOW PUT IT INTO AC5
012462  172605                          LDD     AC5,AC2         ;NOW PUT IT INTO AC2
012464  172200                          ADDD    AC0,AC2         ;ADD 5 TO 5
012466  173205                          SUBD    AC5,AC2         ;SUB 5 FROM 10
012470  175637  001170                  STCDI   AC2,@#$REGO     ;PUT ANS INTO @#$REGO
012474  022737  000005  001170          CMP     #5,@#$REGO      ;WERE THE TWO AC'S EQUAL
012502  170267  166756                  STFPS   FPS             ;GET STATUS
012506  001401                          BEQ     .+4             ;YES
012510  104012                          ERROR   12              ;$REGO SHOULD = 5
```

```
2989                                    ;;*****************************************************************
2990                                    ;*TEST 64        MULD AND DIVD TEST
2991                                    ;;*****************************************************************
2992  012512  000004                    TST64:  SCOPE
2993  012514  012702  001200                    MOV     #$REG4,R2       ;ADDR(DATA)
2994  012520  177227  000052                    LDCID   #52,AC2         ;LOAD AC2 WITH FLOATING DOUBLE 52
2995  012524  174267  166450                    STD     AC2,$REG4       ;PUT IT INTO $REG4
2996  012530  177327  000025                    LDCID   #25,AC3         ;LOAD AC3 WITH FL DB 25
2997  012534  171312                            MULD    (R2),AC3        ;MUL 52X25 RESULT IN AC3
2998  012536  174712                            DIVD    (R2),AC3        ;DIV 52 INTO RESULT IN AC3
2999  012540  171312                            MULD    (R2),AC3        ;MUL 52X25
3000  012542  172312                            ADDD    (R2),AC3        ;ADD AC4 TO AC3 TO MAKE 53 TIMES
3001  012544  012704  000025                    MOV     #25,R4          ;SET UP COUNTER
3002  012550  173367  166424           SUBT:    SUBD    $REG4,AC3       ;SUB 52 FROM AC3 25 TIMES
3003  012554  005304                            DEC     R4
3004  012556  001374                            BNE     SUBT            ;SUB 53 TIMES
3005  012560  175737  001170                    STCDI   AC3,@#$REGO     ;ANS SHOULD BE 52
3006  012564  170267  166674                    STFPS   FPS             ;GET STATUS
3007  012570  022737  000052  001170            CMP     #52,@#$REGO     ;IS ANS CORRECT?
3008  012576  001401                            BEQ     .+4             ;YES
3009  012600  104012                            ERROR   12              ;$REGO SHOULD BE 52
```

```
                                ;;*****************************************************************
                                ;*TEST 65        EXERCISER TEST
                                ;;*****************************************************************
                        TST65:  SCOPE
                                LDCID   #25252,AC0      ;LOAD 25252 INTO AC0
                                LDCID   #25,AC1         ;LOAD AC1 WITH 25
                                MULD    AC0,AC1         ;MUL 25252X25 ANS IN AC1
                                STD     AC1,@#$REG0     ;SAV ANS IN @#$REG0
                                MOV     #24,R2          ;SET UP COUNT
                                STD     AC0,AC5         ;PUT 25252 INTO AC5
                        AAD:    ADDD    AC5,AC0         ;ADD 25252 TO 25252
                                DEC     R2              ;DO 25 TIMES
                                BNE     AAD             ;DONE?
                                STD     AC0,@#$REG4     ;LOAD TO PRINT
                                STFPS   FPS             ;GET STATUS
                                CMPD    @#$REG0,AC0     ;IS ANS CORRECT?
                                CFCC
                                BEQ     .+4             ;YES
                                ERROR   10              ;EITHER THE ADD OR THE MUL DID NOT WORK
                                LDCID   #2,AC3          ;LOAD AC3 WITH A 2
                                MOV     #13,R2          ;SET UP COUNTER
                                STD     AC3,AC4         ;LOAD AC4 WITH A 2
                        MMUL:   MULD    AC4,AC3         ;MUL 2 X 2
                                DEC     R2              ;DO 16 TIMES
                                BNE     MMUL
                                STCDI   AC3,R3          ;PUT ANS IN R3
                                MOV     R3,@#$REG0      ;NOW PUT IT INTO @#$REG0
                                CMP     #10000,R3       ;ANS SHOULD BE 10000
                                STFPS   FPS             ;GET STATUS
                                BEQ     .+4             ;CONT. IF ANS IS CORRECT
                                ERROR   12              ;ANS SHOULD BE 10000
                                LDCID   #10000,AC2      ;LOAD AC2 WITH 10000
                                LDCID   #2,AC1          ;LOAD AC1 WITH A 2
                                MOV     #13,R2          ;SET UP COUNTER
                        DDIV:   DIVD    AC1,AC2         ;DIVD 2 INTO 65536 16 TIMES
                                DEC     R2              ;COUNT NO OF TIMES
                                BNE     DDIV            ;ARE WE DONE?
                                STCDI   AC2,R3          ;STOR ANS INTO R3
                                MOV     R2,@#$REG0      ;PUT IT INTO @#$REG0 FOR TYPING
                                CMP     #2,R3           ;IS ANS CORRECT
                                STFPS   FPS             ;GET STATUS
                                BEQ     .+4             ;ANS IS CORRECT
                                ERROR   12              ;$REG0 SHOULD EQUAL =2
```

```
                                        ;;*****************************************************************
                                        ;*TEST 66        MODE ONE TEST
                                        ;;*****************************************************************
013766  000004                  TST66:  SCOPE
013770  170127  047400                  LDFPS   #47400
013774  177327  000005                  LDCIF   #5,AC3          ;AC3=5
014000  012702  001170                  MOV     #$REGO,R2       ;R2=$REGO
014004  177227  000007                  LDCIF   #7,AC2          ;LOAD 7 INTO AC2
014010  174322                          STF     AC3,(R2)+       ;$REGO SHOULD =5=R2
014012  173737  001170                  CMPF    @#$REGO,AC3     ;DOES @#$REGO = 5
014016  170267  166442                  STFPS   FPS             ;GET STATUS
014022  170000                          CFCC
014024  001401                          BEQ     .+4             ;YES BRANCH
014026  104011                          ERROR   11              ;$REGO SHOULD CONTAIN 5
014030  171242                          MULF    -(R2),AC2       ;MUL 5 X 7,AC2 = 35
014032  012703  000006                  MOV     #6,R3           ;SET UP COUNTER
014036  173212                  SUBFM:  SUBF    (R2),AC2        ;SUB 5 FROM 35
014040  005303                          DEC     R3              ;DO 7 TIMES
014042  001375                          BNE     SUBFM
014044  175637  001170                  STCFI   AC2,@#$REGO     ;@#$REGO SHOULD =5
014050  022737  000005  001170          CMP     #5,@#$REGO      ;DOES @#$REGO =5
014056  170267  166402                  STFPS   FPS             ;GET STATUS
014062  001401                          BEQ     .+4             ;BRANCH IF YES
014064  104012                          ERROR   12              ;ANS SHOULD =5
                                        ;;*****************************************************************
                                        ;*TEST 67        ADDD EXERCISER
                                        ;;*****************************************************************
013066  000004                  TST67:  SCOPE
013070  170127  047600                  LDFPS   #47600
013074  177027  000252                  LDCID   #252,AC0        ;LOAD AC0 WITH 252
013100  177227  052525                  LDCID   #52525,AC2      ;LOAD AC2 WITH 52525
013104  174204                          STD     AC2,AC4         ;AC4=52525
013106  171002                          MULD    AC2,AC0         ;AC0=52525 X 252
013110  012702  000251                  MOV     #251,R2         ;SET UP COUNTER
013114  172204                  AADDM:  ADDD    AC4,AC2         ;ADD 52525 TO 252
013116  005302                          DEC     R2              ;DO 252 TIMES
013120  001375                          BNE     AADDM           ;DONE?
013122  174237  001170                  STD     AC2,@#$REGO     ;GET FOR PRINTING
013126  170267  166332                  STFPS   FPS             ;GET STATUS
013132  173402                          CMPD    AC2,AC0         ;DOES AC2 = AC0?
013134  170000                          CFCC
013136  001401                          BEQ     .+4             ;BRANCH IF EQUAL
013140  104010                          ERROR   10              ;ANS SHOULD BE.....
```

```
                                        ;;*****************************************************************
                                        ;*TEST 70        TEST DIVIDE BY 0
                                        ;;*****************************************************************
013142  000004                  TST70:  SCOPE
013144  170127  047600                  LDFPS   #47600
013150  177027  077777                  LDCID   #77777,AC0
013154  177127  000000                  LDCID   #0,AC1
013160  174401                          DIVD    AC1,AC0         ;DIVIDE 0 INTO 77777
013162  175437  001170                  STCDI   AC0,@#$REG0     ;LOAD @#$REG0 WITH 77777
013166  170267  166272                  STFPS   FPS             ;GET STATUS
013172  170337  001466                  STST    @#FEC           ;GET @#FEC STATUS
013176  022737  077777  001170          CMP     #77777,@#$REG0
013204  001401                          BEQ     .+4
013206  104012                          ERROR   12              ;ANS SHOULD =77777
013210  022737  000004  001466          CMP     #4,@#FEC        ;DID WE TRY TO DIV BY 0?
013216  001401                          BEQ     .+4             ;YES
013220  104002                          ERROR   2               ;FEC SHOULD =4

                                        ;;*****************************************************************
                                        ;*TEST 71        EXERCISER FOR ADDD,SUBD,MULD AND DIVD
                                        ;;*****************************************************************
013222  000004                  TST71:  SCOPE
013224  170127  047400                  LDFPS   #47400
013230  170011                          SETD                    ;SET DOUBLE MODE
013232  177027  077777                  LDCID   #77777,AC0      ;LOAD AC0 WITH 77777
013236  177127  002525                  LDCID   #2525,AC1       ;LOAD AC1 WITH 2525
013242  012702  000012                  MOV     #12,R2          ;SET UP COUNTER
013246  174401                  EXLOP:  DIVD    AC1,AC0         ;DIVDE 2525 INTO 77777
013250  171001                          MULD    AC1,AC0         ;MUL 2525 X ANS
013252  172001                          ADDD    AC1,AC0         ;ADD 2525 TO ANS
013254  173001                          SUBD    AC1,AC0         ;SUB 2525 FROM ANS
013256  005302                          DEC     R2              ;DO 12 TIMES
013260  001372                          BNE     EXLOP           ;DONE?
013262  175437  001170                  STCDI   AC0,@#$REG0     ;LOAD ANSWER INTO @#$REG0
013266  170267  166172                  STFPS   FPS             ;GET STATUS
013272  022737  077777  001170          CMP     #77777,@#$REG0  ;IS @#$REG0 CORRECT
013300  001401                          BEQ     .+4             ;BRANCH IF CORRECT
013302  104012                          ERROR   12
013304  172437  001556          EX4:    LDD     @#D1010,AC0     ;GET DATA
013310  012702  001170                  MOV     #$REG0,R2
013314  174022                          STD     AC0,(R2)+
013316  172537  001556                  LDD     @#D1010,AC1
013322  174122                          STD     AC1,(R2)+
013324  172737  001200                  LDD     @#$REG4,AC3
013330  170267  166130                  STFPS   FPS             ;GET STATUS
013334  173737  001170                  CMPD    @#$REG0,AC3
013340  170000                          CFCC
013342  001401                          BEQ     .+4
013344  104010                          ERROR   10              ;$REG0 AND $REG4 SHOULD =1010
013346  172537  001560                  LDD     @#D0101,AC1
013352  174142                          STD     AC1,-(R2)       ;STORED AC1 IN $REG4-7
013354  172012                          ADDD    (R2),AC0
013356  173042                          SUBD    -(R2),AC0
013360  173001                          SUBD    AC1,AC0
013362  175437  001170                  STCDI   AC0,@#$REG0
013366  022737  000000  001170          CMP     #0,@#$REG0
```

```
3153  013374  170267  166064                    STFPS   FPS             ;GET STATUS
3154  013400  001401                            BEQ     .+4
3155  013402  104012                            ERROR   i2
3156  013404  170001                    MORE:   SETF                    ;SET FLOATING MODE
3157  013406  177027  000525                    LDCIF   #525,AC0        ;LOAD AC0 WITH 525
3158  013412  177127  000252                    LDCIF   #252,AC1        ;LOAD AC1 WITH 252
3159  013416  174104                            STF     AC1,AC4         ;LOAD AC4 WITH 252
3160  013420  172104                            ADDF    AC4,AC1         ;ADD 252 TO 252
3161  013422  172701                            LDF     AC1,AC3         ;PUT ANS IN AC3=524
3162  013424  173003                            SUBF    AC3,AC0         ;SUB 524 FROM 525
3163  013426  175037  001170                    STEXP   AC0,@#$REG0     ;1 IN $REG0
3164  013432  170267  166026                    STFPS   FPS             ;GET STATUS
3165  013436  022737  000001  001170            CMP     #1,@#$REG0      ;CORRECT  ANS SHOULD BE 1
3166  013444  001401                            BEQ     .+4
3167  013446  104012                            ERROR   i2              ;ANS SHOULD BE 5
3168  013450  177027  000021                    LDCIF   #21,AC0         ;LOAD AC0 WITH 21
3169  013454  171000                            MULF    AC0,AC0         ;21 TIMES 21 AC0 = 441
3170  013456  174427  040400                    DIVF    #2,AC0          ;DIV BY 2
3171  013462  012701  001552                    MOV     #FLTONE+4,R1    ;ADDR(DATA)+4
3172  013466  171441                            MODF    -(R1),AC0       ;MUL 1 * 441.
3173  013470  170267  165770                    STFPS   FPS             ;GET STATUS
3174  013474  175537  001170                    STCFI   AC1,@#$REG0     ;PUT $REG0IN $REG0
3175  013500  022737  000220  001170            CMP     #220,@#$REG0    ;IS IT EQUAL?
3176  013506  001401                            BEQ     .+4             ;YES
3177  013510  104012                            ERROR   i2              ;SHOULD = 220
3178  013512  171427  041040                    MODF    #10.,AC0        ;GET FRACTION
3179  013516  175537  001170                    STCFI   AC1,@#$REG0     ;PUT AC0 INTO $REG0
3180  013522  022737  000005  001170            CMP     #5,@#$REG0      ;$REG0 SHOULD = 5
3181  013530  001401                            BEQ     .+4
3182  013532  104012                            ERROR   i2              ;$REG0 SHOULD = 5
3183
3184
3185
3186                                    ;*************************************************************************
3187
3188                                            ;*DONE WITH BASIC INSTRUCTION EXERCISER
3189                                            ;*CLEAR ALL LINE CLOCKS ENABLED IN SYSTEM
3190
3191  013534  105767  165756                    TSTB    CLKPRS          ;KW11-L PRESENT ?
3192  013540  100002                            BPL     NKWL2           ;NO
3193  013542  005037  177546                    CLR     @#LKS           ;YES, CLEAR IT
3194
3195  013546  005767  165744            NKWL2:  TST     CLKPRS          ;KW11-P PRESENT ?
3196  013552  100002                            BPL     NKWP2           ;NO
3197  013554  005037  172540                    CLR     @#PLKS          ;YES, CLEAR IT
3198  013560                            NKWP2:
3199
```

```
                                ;;*****************************************************************
                                        .ENABL  AMA             ;ASSEMBEL ALL PC RELATIVE REFERENCES AS ABSOLUTE
                                ;;*****************************************************************

                                ;;*****************************************************************
                                .SBTTL  SUB PASS END CONTROL

013560  000004                         SCOPE           ;CHECK FOR TEST ITERATIONS HERE

                                ;IF TEST ONLY EITHER HFP OR WFP, ENTER "EOP" ROUTINE DIRECTLY

                                ; IF IN ALTERNATE HFP/WFP MODE,
                                ;   COMPLEMENT FLAG<5>, HFP ENABLE BIT,
                                ;   ENTER EOP ROUTINE ONLY IF ABOUT TO TEST HFP NEXT,
                                ;       TESTING SEQUENCE IS:  PASS#1  HFP SUB-PASS
                                ;                             PASS#1  WFP SUB-PASS
                                ;                             PASS#2  HFP SUB-PASS
                                ;                               ...
013562  076600  000022                 MED     ,RWHAMI         ;GET WHAMI INTO R0.
013566  032700  000020                 BIT     #BIT04,R0       ;1=HFP PRESENT, 0=NONE
013572  001423                         BEQ     $EOP            ;EXIT IF NONE

013574  032777  000002  165344         BIT     #SW01,@SWR      ;1=HFP OR WFP TEST ONLY
013602  001017                         BNE     $EOP            ;0=ALTERNATE HFP AND WFP TESTS

013604  012701  010000                 MOV     #BIT12,R1       ;HFP PRESENT, AND IN ALTERNATE MODE;
013610  076600  000144                 MED     ,RFLAG          ;SO READ FLAGS,
013614  030100                         BIT     R1,R0           ;COMPLEMENT FLAG<5>=BIT12=HFP ENABLE FLAG
013616  001402                         BEQ     1$              ;
013620  040100                         BIC     R1,R0           ;CLEAR BIT 12
013622  000401                         BR      2$              ;
013624  050100                 1$:     BIS     R1,R0           ;SET BIT 12
013626  076600  000344         2$:     MED     ,WFLAG          ;REWRITE FLAGS

013632  030100                         BIT     R1,R0           ;HFP OR WFP NEXT ?
013634  001002                         BNE     $EOP            ;IF HFP AGAIN, START NEW PASS
013636  000137  002756                 JMP     @#SUBPAS        ;IF WFP, NEXT SUBPASS

                                ;;*****************************************************************

                                .SBTTL  END OF PASS ROUTINE (MODIFIED SYSMAC)

                                ;*INCREMENT THE PASS NUMBER ($PASS)
                                ;*IF SW<10>=0, DING BELL ON PASS END
                                ;*IF THERE'S A MONITOR, GO TO IT
                                ;*  ELSE JUMP TO NEWPAS

013642                          $EOP:                           ;
013642  005037  001104                 CLR     $ERFLG          ;ZERO ERROR COUNT
013646  005037  001102                 CLR     $TSTNM          ;ZERO TEST NUMBER
013652  005037  001230                 CLR     $TIMES          ;ZERO NUMBER OF ITERATIONS
013656  005237  001252                 INC     $PASS           ;INCREMENT PASS COUNT,
013662  042737  100000  001252         BIC     #100000,$PASS   ;  BUT NEVER LET IN GO NEGATIVE
```

```
3256  013670  005327                    DEC     (PC)+           ;PASS LOOP ?
3257  013672  000001          $EOPCT:   .WORD   1               ;
3258  013674  003021                    BGT     $DOAGN          ;YES
3259  013676  012737                    MOV     (PC)+,@(PC)+    ;RESTORE COUNTER
3260  013700  000001          $ENDCT:   .WORD   1               ;
3261  013702  013672                    $EOPCT                  ;
3262  013704  032777  002000  165234    BIT     #SW10,@SWR      ;BELL ON PASS END ?
3263  013712  001002                    BNE     $GET42          ;NO
3264  013714  104401  001234            TYPE    ,$BELL          ;YES
3265
3266  013720  013700  000042  $GET42:  MOV     @#42,R0         ;GET MONITOR ADDRESS
3267  013724  001405                    BEQ     $DOAGN          ;NO MONITOR
3268  013726  000005                    RESET                   ;CLEAR WORLD
3269
3270  013730  004710          $ENDAD:   JSR     PC,(R0)         ;GO TO MONITOR
3271  013732  000240                    NOP                     ;
3272  013734  000240                    NOP                     ;RESERVED FOR ACT11
3273  013736  000240                    NOP                     ;
3274
3275  013740  000137  002720  $DOAGN:   JMP     @#NEWPAS        ;RETURN
3276
3277
3278
```

```
                                .SBTTL  TRAP/FEC TESTER SUBR

014744  013746  000244         TRPTST: MOV     @#FPPVEC,-(SP)          ; SAVE OLD FPPVEC PC
014750  013746  000246                 MOV     @#FPPVEC+2,-(SP)        ; AND PS
014754  012737  014052  000244         MOV     #FPPTRP,@#FPPVEC        ; NEW VECTOR PC FOR TEST
014762  016537  000002  000246         MOV     2(R5),@#FPPVEC+2        ; NEW VECTOR PS FOR TEST
014770  012700  000006                 MOV     #6,R0           ; LOAD $TMP0-5
014774  010501                         MOV     R5,R1           ;   WITH TEST DATA SETS
014776  012702  001210                 MOV     #$TMP0,R2       ;   FOR DISPLAY LATER
015002  012122                         MOV     (R1)+,(R2)+     ;
015004  077002                         SOB     R0,.-2          ;
015006  012737  014016  001114         MOV     #FLPERR,$LPERR  ; ERROR LOOP TO HERE
015014  010602                         MOV     SP,R2           ; SAVE GOOD SP

015016  012700  000010         FLPERR: MOV     #8.,R0          ; DELAY CTR, 8(10) INSTR EXEC
015022  010206                         MOV     R2,SP           ; RESET TO GOOD SP
015024  170165  000006                 LDFPS   6(R5)           ; SET UP INITIAL FPS
015030  011537  177776                 MOV     (R5),@#PSW      ; SET UP INITIAL PSW
                                                                ; NOTE:  THE "LOADED PS" IS SETUP WITH PROCESSOR
                                                                ;       PRIORITY = (7), SO THAT INTERRUPTS FROM
                                                                ;       ANY CLOCKS ENABLED ARE EFFECTIVELY
                                                                ;       LOCKED OUT DURING THIS TESTING
                                                                ;       ROUTINE.

015034  000114                         JMP     (R4)            ; GO TEST W/O USING STACK

                                ;-----  NO TRAP RETURNS HERE  ----------------------------------------
015036  032765  040000  000012 FPPNTP: BIT     #BIT14,12(R5)   ; NO TRAP EXPECTED ?
015044  001455                         BEQ     FCONT1          ; BR IF YES
015046  104014                         ERROR   14              ; DIDNT TRAP, SHOULD HAVE

015050  000453                         BR      FCONT1          ; CONTINUE WITH TEST

                                ;-----  TRAP RETURNS HERE  -------------------------------------------
015052  013737  177776  001512 FPPTRP: MOV     @#PSW,NPSLOD    ; SAVE LOADED PSW
015060  011637  001506                 MOV     (SP),OPCRCV     ; GET PUSHED PC
015064  016637  000002  001510         MOV     2(SP),OPSRCV    ; GET PUSHED PS
015072  010637  001514                 MOV     SP,OSPRCV       ; GET SP AFTER TRAP
015076  012716  014104                 MOV     #2$,(SP)        ; FUDGE RETURN
015102  000002                         RTI                     ; RETURN

015104  032765  040000  000012 2$:     BIT     #BIT14,12(R5)   ; TRAP EXPECTED ?
015112  001001                         BNE     1$              ; BR IF YES
015114  104015                         ERROR   15              ; TRAPPED, SHOULDNT HAVE

015116  010237  001504         1$:     MOV     R2,EXPSP        ; GENERATE EXPECTED SP CONTENTS
015122  162737  000004  001504         SUB     #4,EXPSP        ;   AFTER TRAP
015130  023737  001514  001504         CMP     OSPRCV,EXPSP    ; IS SP WHERE IT SHOULD BE?
015136  001401                         BEQ     64$             ;
015140  104016                         ERROR   16              ; NOT EQUAL, SIGNAL ERROR
015142                          64$:

015142  023737  001502  001506         CMP     EXPRTS,OPCRCV   ; CHECK STORED PC IS OK
015150  001401                         BEQ     65$             ;
015152  104017                         ERROR   17              ; NOT EQUAL, SIGNAL ERROR
015154                          65$:
```

```
3335  014154  026537  000004  001510          CMP     4(R5),OPSRCV    ; CHECK STORED PS IS OK
3336  014162  001401                          BEQ     66$             ;
3337  014164  104020                          ERROR   20              ; NOT EQUAL, SIGNAL ERROR
3338  014166                          66$:
3339
3340  014166  026537  000002  001512          CMP     2(R5),NPSLOD    ; CHECK LOADED PS IS OK
3341  014174  001401                          BEQ     67$             ;
3342  014176  104021                          ERROR   21              ; NOT EQUAL, SIGNAL ERROR
3343  014200                          67$:
3344
3345                                  ;-----  CONTINUE  -----
3346  014200  170237  001464          FCONT1: STFPS   FPS             ; STORE FPS AFTER
3347  014204  170337  001466                  STST    FEC             ; STORE FEC/FEA AFTER
3348
3349  014210  023765  001464  000010          CMP     FPS,10(R5)      ; CHECK FPS
3350  014216  001401                          BEQ     65$             ; FPS IS OK
3351  014220  104023                          ERROR   23              ; FPS BAD
3352  014222  005765  000012          65$:    TST     12(R5)          ; DOES FEC/FEA APPLY?
3353  014226  100014                          BPL     66$             ; NO - SKIP TEST
3354  014230  013737  001500  001500          MOV     EXPFEA,EXPFEA   ; GET EXPECTED FEA
3355  014236  123765  001466  000012          CMPB    FEC,12(R5)      ; COMPARE FEC-S
3356  014244  001004                          BNE     64$             ; NOT EQUAL
3357  014246  023737  001470  001500          CMP     FEA,EXPFEA      ; COMPARE FEA-S
3358  014254  001401                          BEQ     66$             ; FEC, FEA OK
3359  014256  104022                  64$:    ERROR   22              ; FEC OR FEA ARE BAD
3360  014260                          66$:
3361
3362  014260  010206                          MOV     R2,SP           ; RESTORE GOOD SP AFTER TEST, IN CASE
3363
3364  014262  012637  000246                  MOV     (SP)+,@#FPPVEC+2        ; RESET STANDARD FPP VECTOR
3365  014266  012637  000244                  MOV     (SP)+,@#FPPVEC          ;
3366
3367  014272  005037  177776                  CLR     @#PSW           ; SETUP FOR INTERRUPT ENABLE, PRO
3368  014276  000207                          RTS     PC              ; RETURN TO TEST CALLER
3369
3370                                  ;-----  LOCAL FEC-CODE GENERATING SUBR  ------------------------------
3371
3372  014300                          FEC02:                          ;
3373  014300  170406                  FII02:  170406                  ; ILLEGAL FPP INSTR
3374  014302  077001                  NXT02:  SOB     R0,.            ; DELAY FOR FPP TO SETTLE
3375  014304  000137  014036                  JMP     FPPNTP          ;
3376
3377  014310  172437  014442          FEC04:  LDF     FLT001,AC0      ;
3378  014314  174437  014432          FII04:  DIVF    FLTZER,AC0      ; X/0.0 = ?
3379  014320  077001                  NXT04:  SOB     R0,.            ; DELAY FOR FPP TO SETTLE
3380  014322  000137  014036                  JMP     FPPNTP          ;
3381
3382  014326  172537  014442          FEC06:  LDF     FLT001,AC1      ;
3383  014332  175537  001170          FII06:  STCFI   AC1,$REG0       ; FLT001 > LGST I
3384  014336  077001                  NXT06:  SOB     R0,.            ; DELAY FOR FPP TO SETTLE
3385  014340  000137  014036                  JMP     FPPNTP          ;
3386
3387  014344  172637  014442          FEC10:  LDF     FLT001,AC2      ;
3388  014350  172737  014446                  LDF     FLT002,AC3      ;
3389  014354  171203                  FII10:  MULF    AC3,AC2         ; LG * LG = OVFLW
3390  014356  077001                  NXT10:  SOB     R0,.            ; DELAY FOR FPP TO SETTLE
```

```
3391  014360  000137  014036                JMP     FPPNTP          ;
3392
3393  014364  172637  014452    FEC12:      LDF     FLT003,AC2      ;
3394  014370  171237  014456    FII12:      MULF    FLT004,AC2      ; SM * SM = UNDFLW
3395  014374  077001             NXT12:      SOB     R0,.            ; DELAY FOR FPP TO SETTLE
3396  014376  000137  014036                JMP     FPPNTP          ;
3397
3398  014402                     FEC14:                              ;
3399  014402  172737  014436    FII14:      LDF     FLTMZR,AC3      ; -0.0 -> AC3
3400  014406  077001             NXT14:      SOB     R0,.            ; DELAY FOR FPP TO SETTLE
3401  014410  000137  014036                JMP     FPPNTP          ;
3402
3403  014414  012703  000342    FEC16:      MOV     #342,R3         ;HFP U-ADDR FOR "LDF/D" INSTR
3404  014420  170003                        LDUB    R3,R3           ;
3405  014422  172400             FII16:      LDF     AC0,AC0         ; U-BREAK TRAP
3406  014424  077001             NXT16:      SOB     R0,.            ; DELAY FOR FPP TO SETTLE
3407  014426  000137  014036                JMP     FPPNTP          ;
3408
3409                             ;-----  CONSTANTS  ----------------------------------------------------
3410  014432  000000  000000    FLTZER:     .WORD   000000,000000   ; +0.0
3411  014436  100000  000000    FLTMZR:     .WORD   100000,000000   ; -0.0
3412  014442  044000  000000    FLT001:     .WORD   044000,000000   ; .1E+20 = 65536
3413  014446  074377  177777    FLT002:     .WORD   074377,177777   ; .111 ... 1E+161
3414                             ; NOTE FLT001*FLT002 = .111 ... 1E+200  (OVERFLOW)
3415  014452  030000  000000    FLT003:     .WORD   030000,000000   ; .1E-40
3416  014456  010000  000000    FLT004:     .WORD   010000,000000   ; .1E-140
3417                             ; NOTE FLT003*FLT004 = .1E-201  (UNDERFLOW)
```

```
3418                                        .SBTTL  LINE CLOCK INTERRUPT SERVICE ROUTINES
3419
3420  014462  005037  177546          IKW11L: CLR     @#LKS           ;CLEAR STATUS
3421  014466  005237  001520                  INC     KW11LC          ;BUMP COUNTER
3422  014472  042737  100000  001520          BIC     #BIT15,KW11LC   ;NEVER OVERFLOW
3423  014500  052737  000100  177546          BIS     #BIT6,@#LKS     ;RESTART CLOCK
3424  014506  000002                          RTI                     ;AND RETURN
3425
3426  014510  005037  172540          IKW11P: CLR     @#PLKS          ;CLEAR STATUS
3427  014514  005237  001522                  INC     KW11PC          ;BUMP COUNTER
3428  014520  042737  100000  001522          BIC     #BIT15,KW11PC   ;NEVER OVERFLOW
3429
3430                                          ;THE NEXT 13 INSTRUCTIONS GENERATE A 10 BIT RANDOM
3431                                          ;INTEGER FROM 0001-1777 INCLUSIVE TO FEED INTO THE
3432                                          ;KW11-P COUNT SET REGISTER.
3433  014526  005046                          CLR     -(SP)           ;TEMP STORAGE FOR SHIFTED OUT BIT
3434  014530  006237  001524                  ASR     KW11PR          ;SFT OLD CONTENTS RITE 1; LOB INTO C
3435  014534  006116                          ROL     (SP)            ;PUT OLD LOB ON STACK
3436  014536  032737  000004  001524          BIT     #BIT2,KW11PR    ;WAS OTHER BIT ON ?
3437  014544  001403                          BEQ     1$              ;NO
3438  014546  005116                          COM     (SP)            ;YES, COMPLEMENT OLD LOB
3439  014550  042716  177776                  BIC     #177776,(SP)    ;ELIMINATE JUNK
3440  014554  000316                  1$:     SWAB    (SP)            ;FAST SHIFT LOB TO BIT9 POSITION
3441  014556  006316                          ASL     (SP)            ;
3442  014560  052637  001524                  BIS     (SP)+,KW11PR    ;PUT NEW HOB INTO COUNTER
3443  014564  042737  176000  001524          BIC     #176000,KW11PR  ;NO HIGH ORDER JUNK
3444  014572  001003                          BNE     2$              ;ZERO IS AN ILLEGAL STATE,
3445  014574  012737  001777  001524          MOV     #1777,KW11PR    ;  SO RESET TO START
3446
3447  014602  013737  001524  172542  2$:     MOV     KW11PR,@#PLKR   ;SET NEW COUNT
3448  014610  052737  000101  172540          BIS     #BIT6+BIT0,@#PLKS       ;RESTART CLOCK
3449  014616  000002                          RTI                     ;AND RETURN
```

```
                                .SBTTL  FPP TRAP CATCHER

014620 010637 001514    FPPILT: MOV     SP,OSPRCV       ;SP AFTER TRAP
014624 012637 001472            MOV     (SP)+,FPPOPC    ;POP OLD PC FOR DISPLAY
014630 012637 001474            MOV     (SP)+,FPPOPS    ;POP OLD PS FOR DISPLAY
014634 170237 001464            STFPS   FPS             ;GET FPS
014640 170337 001466            STST    FEC             ;GET FEC/FEA
014644 005737 001464            TST     FPS             ;TEST ERROR BIT
014650 100007                   BPL     1$              ;OFF - NO ERROR BIT SET, BUT TRAPPED
                                                        ;ON  - IT SHOULD BE ON A TRAP
014652 032737 040000 001464     BIT     #040000,FPS     ;TEST INTERRUPT ENABLE BIT
014660 001003                   BNE     1$              ;ON  - INTR DISABLED, BUT TRAPPED
                                                        ;OFF - ABLE TO INTR, SO IGNORE IT,
014662 012701 177777                    MOV     #-1,R1  ;       BUT FLAG THAT IT OCCURRED
014666 000401                   BR      2$              ;       AND SKIP THE ERROR
014670 104024           1$:     ERROR   24              ;SIGNAL UNEXPECTED FPP TRAP
014672 013746 001474    2$:     MOV     FPPOPS,-(SP)    ;PUSH PSW
014676 013746 001472            MOV     FPPOPC,-(SP)    ;PUSH PC
014702 000002                   RTI                     ;CONTINUE,  RECOVER AT LAST TRAP ONLY
```

```
                                .SBTTL  SCOPE HANDLER ROUTINE

                                ;;**********************************************************************
                                ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
                                ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<15:0>)
                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                ;*SW14=1         LOOP ON TEST
                                ;*SW11=1         INHIBIT ITERATIONS
                                ;*SW09=1         LOOP ON ERROR
                                ;*SW08=1         LOOP ON TEST IN "$LPTST"
                                ;*CALL
                                ;*      SCOPE           ;;SCOPE=IOT

014704                          $SCOPE:
014704                          64$:
014704  032777  040000  164234  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
014712  001114                          BNE     $OVER           ;;YES IF SW14=1
                                ;#####START OF CODE FOR THE XOR TESTER#####
014714  000416                  $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
014716  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
014722  012737  014742  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
014730  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
014734  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
014740  000463                          BR      $SVLAD          ;;GO TO THE NEXT TEST
014742  022626                  5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
014744  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
014750  000423                          BR      7$              ;;LOOP ON THE PRESENT TEST
014752                          6$:;#####END OF CODE FOR THE XOR TESTER#####
014752  032777  000400  164166          BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
014760  001404                          BEQ     2$              ;;BR IF NO
014762  023737  001112  001102          CMP     $LPTST,$TSTNM   ;;ON THE RIGHT TEST?
014770  001465                          BEQ     $OVER           ;;BR IF YES
014772  005737  001104          2$:     TST     $ERFLG          ;;HAS AN ERROR OCCURRED?
014776  001421                          BEQ     3$              ;;BR IF NO
015000  023737  001122  001104          CMP     $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
015006  101015                          BHI     3$              ;;BR IF NO
015010  032777  001000  164130          BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
015016  001404                          BEQ     4$              ;;BR IF NO
015020  013737  001114  001110  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
015026  000446                          BR      $OVER
015030  005037  001104          4$:     CLR     $ERFLG          ;;ZERO THE ERROR FLAG
015034  005037  001230                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
015040  000415                          BR      1$              ;;ESCAPE TO THE NEXT TEST
015042  032777  004000  164076  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
015050  001011                          BNE     1$              ;;BR IF YES
015052  005737  001252                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
015056  001406                          BEQ     1$              ;;      INHIBIT ITERATIONS
015060  005237  001106                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
015064  023737  001230  001106          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
015072  002024                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
015074  012737  000001  001106  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
015102  013737  015160  001230          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
015110  005237  001102          $SVLAD: INC     $TSTNM          ;;COUNT TEST NUMBERS
015114  013737  001102  001250          MOV     $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
015122  011637  001110                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
```

```
 3526  015126  011637  001114                MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
 3527  015132  005037  001232                CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
 3528  015136  012737  000001  001122        MOV     #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
 3529  015144  013777  001102  163776 $OVER: MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
 3530  015152  013716  001110               MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
 3531  015156  000002                       RTI                     ;;FIXES PS
 3532  015160  003720              $MXCNT: 2000.                   ;;MAX. NUMBER OF ITERATIONS
```

```
                                .SBTTL  ERROR HANDLER ROUTINE

                                ;;*****************************************************************
                                ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
                                ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
                                ;*AND GO TO $TYPERR ON ERROR
                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                ;*SW15=1         HALT ON ERROR
                                ;*SW13=1         INHIBIT ERROR TYPEOUTS
                                ;*SW10=1         BELL ON ERROR
                                ;*SW09=1         LOOP ON ERROR
                                ;*CALL
                                ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER

015162                          $ERROR:
015162  010037  001526                  MOV     R0,EREG0        ; DISPLAY R0
015166  010137  001530                  MOV     R1,EREG1        ;         R1
015172  010237  001532                  MOV     R2,EREG2        ;         R2
015176  010337  001534                  MOV     R3,EREG3        ;         R3
015202  010437  001536                  MOV     R4,EREG4        ;         R4
015206  010537  001540                  MOV     R5,EREG5        ;         R5
015212  010637  001542                  MOV     R6,EREG6        ; GET R6(SP) BEFORE TRAP
015216  062737  000004  001542          ADD     #4,EREG6        ;
015224  011637  001544                  MOV     (SP),EREG7      ; PC -> ERROR CALL INSTR
015230  005237  001104          7$:     INC     $ERFLG          ;;SET THE ERROR FLAG
015234  001775                  BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
015236  013777  001102  163704  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
015244  032777  002000  163674  BIT     #BIT10,@SWR     ;;BELL ON ERROR?
015252  001402                  BEQ     1$              ;;NO - SKIP
015254  104401  001234          TYPE    ,$BELL          ;;RING BELL
015260  005237  001116          1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
015264  011637  001124          MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
015270  162737  000002  001124  SUB     #2,$ERRPC
015276  117737  163622  001120  MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
015304  032777  020000  163634  BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
015312  001004                  BNE     20$             ;;SKIP TYPEOUTS
015314  004737  015424          JSR     PC,$TYPERR      ;;GO TO USER ERROR ROUTINE
015320  104401  001241          TYPE    ,$CRLF
015324                          20$:
015324  122737  000001  001264  CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
015332  001007                  BNE     2$              ;;NO,SKIP APT ERROR REPORT
015334  113737  001120  015346  MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
015342  004737  016140          JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
015346     000                  21$:    .BYTE   0
015347     000                  .BYTE   0
015350  000777                  22$:    BR      22$             ;;APT ERROR LOOP
015352  005777  163570          2$:     TST     @SWR            ;;HALT ON ERROR
015356  100001                  BPL     3$              ;;SKIP IF CONTINUE
015360  000000                  HALT                    ;;HALT ON ERROR!
015362  032777  001000  163556  3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
015370  001402                  BEQ     4$              ;;BR IF NO
015372  013716  001114          MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
015376  005737  001232          4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
015402  001402                  BEQ     5$              ;;BR IF NONE
015404  013716  001232          MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
015410                          5$:
```

```
 3589  015410  022737  013730  000042          CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
 3590  015416  001001                          BNE     6$              ;;BRANCH IF NO
 3591  015420  000000                          HALT                    ;;YES
 3592  015422                          6$:
 3593  015422  000002                  64$:    RTI                     ;RETURN
```

```
                                ;;*****************************************************************
                                .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE (MODIFIED SYSMAC)

                                ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
                                ;*ERROR IS TO BE REPORTED.  IT THEN OBTAINS, FROM THE "ERROR TABLE",
                                ;*($ERRTB) THE ERROR MESSAGE, DATA HEADER, AND DATA VALUES TO PRINT.
                                ;*THIS ROUTINE IS IDENTICAL TO THE SYSMAC ROUTINE $ERRTYP, EXCEPT THIS
                                ;*ROUTINE PUTS A <HT> BETWEEN OCTAL TYPED DATA VALUES, SO THAT EACH
                                ;*VALUE STARTS AT A HORIZONTAL TAB STOP.  ALSO, THE DATA FORMAT
                                ;*POINTER HAS BEEN ELIMINATED FROM THE ERROR VECTOR.  THIS ROUTINE
                                ;*ALSO ALWAYS PRINTS $TESTN AND $ERRPC AS THE FIRST TWO DATA ELEMENTS
                                ;*(WITH APPROPRIATE HEADERS).

015424                          $TYPERR:
015424  104401                          TYPE                    ; TYPE "HOT" OR "WARM"
015426  001241                  HOTWRM: .WORD   $CRLF            ; PTR TO MESSAGE
015430  010046                          MOV     R0,-(SP)        ; SAVE R0
015432  010146                          MOV     R1,-(SP)        ; SAVE R1
015434  005000                          CLR     R0              ; PICKUP ITEM INDEX
015436  153700  001120                  BISB    @#$ITEMB,R0     ;
015442  001004                          BNE     1$              ; IF ITEM NUMBER FROM ERROR 0,
                                                                ;   JUST TYPE PC OF ERROR
015444  013746  001124                  MOV     $ERRPC,-(SP)    ; GET ERROR PC FOR TYPEOUT
015450  104402                          TYPOC                   ; TYPE OCTAL, ALL DIGITS
015452  000452                          BR      7$              ; EXIT
015454  005300                  1$:     DEC     R0              ; ADJUST ERROR # FOR TABLE INDEX
015456  006300                          ASL     R0              ;   OF 6 BYTES/ENTRY
015460  010001                          MOV     R0,R1           ;
015462  006300                          ASL     R0              ;
015464  060100                          ADD     R1,R0           ;
015466  062700  001274                  ADD     #$ERRTB,R0      ; FORM TABLE PTR
015472  012037  015502                  MOV     (R0)+,2$        ; PICKUP "ERROR MESSAGE" PTR
015476  001404                          BEQ     3$              ; SKIP TYPEOUT IF NULL
015500  104401                          TYPE                    ; TYPE "ERROR MESSAGE"
015502  000000                  2$:     .WORD   0               ; "ERROR MESSAGE" PTR HERE
015504  104401  001241                  TYPE    ,$CRLF          ; CR & LF
015510  104401  015620          3$:     TYPE    ,11$            ; "TEST #    ERR PC" HEADER
015514  012037  015524                  MOV     (R0)+,4$        ; PICKUP "DATA HEADER" PTR
015520  001402                          BEQ     5$              ; SKIP TYPEOUT IF NULL
015522  104401                          TYPE                    ; TYPE "DATA HEADER"
015524  000000                  4$:     .WORD   0               ; "DATA HEADER" PTR HERE
015526  104401  001241          5$:     TYPE    ,$CRLF          ; CR & LF
015532  017746  000054                  MOV     @8$,-(SP)       ; ($TESTN)
015536  104402                          TYPOC                   ; OCTAL W/ LEADING ZEROS
015540  104401  015616                  TYPE    ,10$            ; <HT>
015544  017746  000044                  MOV     @9$,-(SP)       ; ($ERRPC)
015550  104402                          TYPOC                   ; OCTAL W/ LEADING ZEROS
015552  104401  015616                  TYPE    ,10$            ; <HT>
015556  011000                          MOV     (R0),R0         ; PICKUP "DATA TABLE" PTR
015560  001407                          BEQ     7$              ; EXIT IF NULL
015562  013046                  6$:     MOV     @(R0)+,-(SP)    ; SAVE ... FOR TYPEOUT
015564  104402                          TYPOC                   ; TYPE OCTAL, ALL DIGITS
015566  005710                          TST     (R0)            ; ANOTHER NUMBER ?
015570  001403                          BEQ     7$              ; NO - EXIT
015572  104401  015616                  TYPE    ,10$            ; TAB BETWEEN ELEMENTS
```

```
3650  015576  000771                        BR      6$              ; LOOP ON DATA TABLE VECTOR
3651  015600  012601                7$:     MOV     (SP)+,R1        ; RESTORE R1
3652  015602  012600                        MOV     (SP)+,R0        ; RESTORE R0
3653  015604  104401  001241              TYPE    ,$CRLF          ; CR & LF
3654  015610  000207                        RTS     PC              ; RETURN
3655  015612  001250                8$:     .WORD   $TESTN          ;
3656  015614  001124                9$:     .WORD   $ERRPC          ;
3657  015616  000011                10$:    .ASCIZ  <11>            ; <HT>
3658  015620  042524  052123  021440 11$:    .ASCIZ  "TEST # ERR PC "
3659  015626  042411  051122  050040
3660  015634  004503     000
3661          015640                        .EVEN
```

```
                                .SBTTL  TYPE ROUTINE

                                ;;*****************************************************************
                                ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
                                ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
                                ;*NOTE1:         $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
                                ;*NOTE2:         $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
                                ;*NOTE3:         $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
                                ;*
                                ;*CALL:
                                ;*1) USING A TRAP INSTRUCTION
                                ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
                                ;*OR
                                ;*      TYPE
                                ;*      MESADR
                                ;*

015640  105737  001165          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
015644  100002                          BPL     1$              ;;BR IF YES
015646  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
015650  000430                          BR      3$              ;;LEAVE
015652  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
015654  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
015660  122737  000001  001264          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
015666  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
015670  132737  000100  001265          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
015676  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
015700  010037  015710                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
015704  004737  016130                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
015710  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
015712  132737  000040  001265  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
015720  001003                          BNE     60$             ;;YES,SKIP TYPE OUT
015722  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
015724  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
015726  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
015730  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
015732  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
015736  000002                          RTI                     ;;RETURN
015740  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
015744  001430                          BEQ     8$
015746  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
015752  001006                          BNE     5$
015754  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
015756  104401                          TYPE                    ;;TYPE A CR AND LF
015760  001241                          $CRLF
015762  105037  016116                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
015766  000755                          BR      2$              ;;GET NEXT CHARACTER
015770  004737  016052          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
015774  123726  001164          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
016000  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
016002  013746  001162                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                                ;;AND THE NULL CHAR.
016006  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
016012  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
016014  004737  016052                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
016020  105337  016116                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
```

```
3718  016024  000770                        BR      7$              ;;LOOP
3719
3720                                  ;HORIZONTAL TAB PROCESSOR
3721
3722  016026  112716  000040          8$:   MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
3723  016032  004737  016052          9$:   JSR     PC,$TYPEC       ;;TYPE A SPACE
3724  016036  132737  000007  016116        BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
3725  016044  001372                        BNE     9$              ;;TAB STOP
3726  016046  005726                        TST     (SP)+           ;;POP SPACE OFF STACK
3727  016050  000724                        BR      2$              ;;GET NEXT CHARACTER
3728  016052  105777  163100        $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
3729  016056  100375                        BPL     $TYPEC
3730  016060  116677  000002  163072        MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
3731  016066  122766  000015  000002        CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
3732  016074  001003                        BNE     1$              ;;BRANCH IF NO
3733  016076  105037  016116                CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
3734  016102  000406                        BR      $TYPEX          ;;EXIT
3735  016104  122766  000012  000002  1$:   CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
3736  016112  001402                        BEQ     $TYPEX          ;;BRANCH IF YES
3737  016114  105227                        INCB    (PC)+           ;;COUNT THE CHARACTER
3738  016116  000000        $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
3739  016120  000207        $TYPEX: RTS     PC
3740
```

```
                                        .SBTTL  APT COMMUNICATIONS ROUTINE

                                        ;;*****************************************************************
016122  112737  000001  016366  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
016130  112737  000001  016364  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
016136  000403                  BR      $ATYC
016140  112737  000001  016366  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
016146                          $ATYC:
016146  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
016150  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
016152  105737  016364          TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
016156  001450                  BEQ     5$              ;;IF NOT:  BR
016160  122737  000001  001264  CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
016166  001031                  BNE     3$              ;;IF NOT:  BR
016170  132737  000100  001265  BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
016176  001425                  BEQ     3$              ;;IF NOT:  BR
016200  017600  000004          MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
016204  062766  000002  000004  ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
016212  005737  001244          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
016216  001375                  BNE     1$              ;;IF NOT:  WAIT
016220  010037  001260          MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
016224  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
016226  001376                  BNE     2$
016230  163700  001260          SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
016234  006200                  ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
016236  010037  001262          MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
016242  012737  000004  001244  MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
016250  000413                  BR      5$
016252  017637  000004  016276  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
016260  062766  000002  000004  ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
016266  013746  177776          MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
016272  004737  015640          JSR     PC,$TYPE        ;;CALL TYPE MACRO
016276  000000                  4$:     .WORD   0
016300                          5$:
016300  105737  016366          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
016304  001416                  BEQ     12$             ;;IF NOT:  BR
016306  005737  001264          TST     $ENV            ;;RUNNING UNDER APT?
016312  001413                  BEQ     12$             ;;IF NOT:  BR
016314  005737  001244          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
016320  001375                  BNE     11$             ;;IF NOT:  WAIT
016322  017637  000004  001246  MOV     @4(SP),$FATAL   ;;GET ERROR #
016330  062766  000002  000004  ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
016336  005237  001244          INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
016342  105037  016366          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
016346  105037  016365          CLRB    $LFLG           ;;CLEAR LOG FLAG
016352  105037  016364          CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
016356  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
016360  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
016362  000207                  RTS     PC              ;;RETURN
016364     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
016365     000          $LFLG:  .BYTE   0               ;;LOG FLAG
016366     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
            016370                  .EVEN
            000200          APTSIZE=200
            000001          APTENV=001
            000100          APTSPOOL=100
```

```
  3797          000040                  APTCSUP=040
```

```
                                .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE

                                ;;*****************************************************************
                                ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
                                ;*OCTAL (ASCII) NUMBER AND TYPE IT.
                                ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
                                ;*CALL:
                                ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                ;*      TYPOS                   ;;CALL FOR TYPEOUT
                                ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
                                ;*      .BYTE   M               ;;M=1 OR 0
                                ;*                                      ;;1=TYPE LEADING ZEROS
                                ;*                                      ;;0=SUPPRESS LEADING ZEROS
                                ;*
                                ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
                                ;*$TYPOS OR $TYPOC
                                ;*CALL:
                                ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                ;*      TYPON                   ;;CALL FOR TYPEOUT
                                ;*
                                ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
                                ;*CALL:
                                ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                ;*      TYPOC                   ;;CALL FOR TYPEOUT

016370  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
016374  116637  000001  016613          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
016402  112637  016615                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
016406  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
016412  000406                          BR      $TYPON
016414  112737  000001  016613  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
016422  112737  000006  016615          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
016430  112737  000005  016612  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
016436  010346                          MOV     R3,-(SP)        ;;SAVE R3
016440  010446                          MOV     R4,-(SP)        ;;SAVE R4
016442  010546                          MOV     R5,-(SP)        ;;SAVE R5
016444  113704  016615                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
016450  005404                          NEG     R4
016452  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
016456  110437  016614                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
016462  113704  016613                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
016466  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
016472  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
016474  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
016476  000404                          BR      3$              ;;GO DO MSB
016500  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
016502  006105                          ROL     R5
016504  006105                          ROL     R5
016506  010503                          MOV     R5,R3
016510  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
016512  105337  016614                  DECB    $OMODE          ;;TYPE THIS DIGIT?
016516  100016                          BPL     7$              ;;BR IF NO
016520  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
016524  001002                          BNE     4$              ;;TEST FOR 0
016526  005704                          TST     R4              ;;SUPPRESS THIS 0?
016530  001403                          BEQ     5$              ;;BR IF YES
```

```
3854  016532  005204                 4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
3855  016534  052703  000060                 BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
3856  016540  052703  000040         5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
3857  016544  110337  016610                 MOVB    R3,B$           ;;SAVE FOR TYPING
3858  016550  104401  016610                 TYPE    ,B$             ;;GO TYPE THIS DIGIT
3859  016554  105337  016612         7$:     DECB    $OCNT           ;;COUNT BY 1
3860  016560  003347                 BGT     2$              ;;BR IF MORE TO DO
3861  016562  002402                 BLT     6$              ;;BR IF DONE
3862  016564  005204                 INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
3863  016566  000744                 BR      2$              ;;GO DO THE LAST DIGIT
3864  016570  012605         6$:     MOV     (SP)+,R5        ;;RESTORE R5
3865  016572  012604                 MOV     (SP)+,R4        ;;RESTORE R4
3866  016574  012603                 MOV     (SP)+,R3        ;;RESTORE R3
3867  016576  016666  000002  000004        MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
3868  016604  012616                 MOV     (SP)+,(SP)
3869  016606  000002                 RTI                     ;;RETURN
3870  016610      000         B$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
3871  016611      000                 .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
3872  016612      000         $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
3873  016613      000         $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
3874  016614  000000         $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
```

```
                                .SBTTL  TRAP DECODER

                                ;;**********************************************************************
                                ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
                                ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
                                ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
                                ;*GO TO THAT ROUTINE.

016616  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
016620  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
016624  005740                          TST     -(R0)           ;;BACKUP BY 2
016626  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
016630  006300                          ASL     R0              ;;POSITION FOR INDEXING
016632  016000  016652                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
016636  000200                          RTS     R0              ;;GO TO ROUTINE

                                ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO

016640  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
016642  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
016650  000002                          RTI                     ;;RESTORE THE PSW

                                .SBTTL  TRAP TABLE

                                ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
                                ;*BY THE "TRAP" INSTRUCTION.

                                ;       ROUTINE
                                ;       -------
016652  016640                  $TRPAD: .WORD   $TRAP2
016654  015640                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
016656  016414                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
016660  016370                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
016662  016430                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
```

```
                                .SBTTL  POWER DOWN AND UP ROUTINES

                                ;;**********************************************************************
                                ;POWER DOWN ROUTINE
016664  012737  017036  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
016672  012737  000340  000026         MOV     #340,@#PWRVEC+2 ;;PRIO:7
016700  010046                       MOV     R0,-(SP)        ;;PUSH R0 ON STACK
016702  010146                       MOV     R1,-(SP)        ;;PUSH R1 ON STACK
016704  010246                       MOV     R2,-(SP)        ;;PUSH R2 ON STACK
016706  010346                       MOV     R3,-(SP)        ;;PUSH R3 ON STACK
016710  010446                       MOV     R4,-(SP)        ;;PUSH R4 ON STACK
016712  010546                       MOV     R5,-(SP)        ;;PUSH R5 ON STACK
016714  017746  162226               MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
016720  010637  017042               MOV     SP,$SAVR6       ;;SAVE SP
016724  012737  016736  000024       MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
016732  000000                       HALT
016734  000776                       BR      .-2             ;;HANG UP

                                ;;**********************************************************************
                                ;POWER UP ROUTINE
016736  012737  017036  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
016744  013706  017042               MOV     $SAVR6,SP       ;;GET SP
016750  005037  017042               CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
016754  005237  017042         1$:   INC     $SAVR6          ;;WAIT FOR THE INC
016760  001375                       BNE     1$              ;;OF  WORD
016762  011600                       MOV     (SP),R0         ;GET SAVED SWR OFF STACK
016764  076600  000226               MED      226            ;RESTORE SWR CONTENTS
016770  012677  162152               MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
016774  012605                       MOV     (SP)+,R5        ;;POP STACK INTO R5
016776  012604                       MOV     (SP)+,R4        ;;POP STACK INTO R4
017000  012603                       MOV     (SP)+,R3        ;;POP STACK INTO R3
017002  012602                       MOV     (SP)+,R2        ;;POP STACK INTO R2
017004  012601                       MOV     (SP)+,R1        ;;POP STACK INTO R1
017006  012600                       MOV     (SP)+,R0        ;;POP STACK INTO R0
017010  012737  016664  000024       MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
017016  012737  000340  000026       MOV     #340,@#PWRVEC+2 ;;PRIO:7
017024  104401                       TYPE                    ;REPORT THE POWER FAILURE
017026  017044         $PWRMG: .WORD   $POWER          ;;POWER FAIL MESSAGE POINTER
017030  012716                       MOV     (PC)+,(SP)      ;;RESTART AT START
017032  002100         $PWRAD: .WORD   START           ;;RESTART ADDRESS
017034  000002                       RTI
017036  000000         $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
017040  000776                       BR      .-2             ;;  BEFORE THE POWER DOWN WAS COMPLETE
017042  000000         $SAVR6: 0                       ;;PUT THE SP HERE
017044  005015  047520  042527 $POWER: .ASCIZ  <15><12>"POWER"
017052  000122
                                .EVEN
```

```
                                        .SBTTL  ERROR MESSAGES, DATA HEADERS, DATA VECTORS, ETC

                                        ;MESSAGE PREFIXES
017054  047510  035124  000040  ASCHOT: .ASCIZ  "HOT: "
017062  040527  046522  020072  ASCWRM: .ASCIZ  "WARM: "
017070     000

                                        ;ERROR MESSAGES HERE
017071     122  041505  044505  EMA:    .ASCIZ  "RECEIVED FPS IS BAD"
017076  042526  020104  050106
017104  020123  051511  041040
017112  042101     000
017115     122  041505  044505  EMB:    .ASCIZ  "RECEIVED FEC IS BAD"
017122  042526  020104  042506
017130  020103  051511  041040
017136  042101     000
017141     122  041505  044505  EMC:    .ASCIZ  "RECEIVED FEA IS BAD"
017146  042526  020104  042506
017154  020101  051511  041040
017162  042101     000
017165     122  041505  044505  EMD:    .ASCIZ  "RECEIVED PSW CC-S ARE BAD"
017172  042526  020104  051520
017200  020127  041503  051455
017206  040440  042522  041040
017214  042101     000
017217     102  042101  040440  EME:    .ASCIZ  "BAD ADDRESS IN R0"
017224  042104  042522  051523
017232  044440  020116  030122
017240     000
017241     102  042101  051440  EMF:    .ASCIZ  "BAD STACK POINTER"
017246  040524  045503  050040
017254  044517  052116  051105
017262     000
017263     122  051505  046125  EMH:    .ASCIZ  "RESULT OF FPP ARITHMETIC OPERATION IS BAD"
017270  020124  043117  043040
017276  050120  040440  044522
017304  044124  042515  044524
017312  020103  050117  051105
017320  052101  047511  020116
017326  051511  041040  042101
017334     000
017335     122  041505  044505  EMI:    .ASCIZ  "RECEIVED FEC/FEA IS BAD"
017342  042526  020104  042506
017350  027503  042506  020101
017356  051511  041040  042101
017364     000
017365     125  042516  050130  EMJ:    .ASCIZ  "UNEXPECTED FPP TRAP, IGNORED AND CONTINUING"
017372  041505  042524  020104
017400  050106  020120  051124
017406  050101  020054  043511
017414  047516  042522  020104
017422  047101  020104  047503
017430  052116  047111  044525
017436  043516     000
017441     103  052520  042040  EMP:    .ASCIZ  "CPU DID NOT TRAP ON FPP EXCEPTION, BUT IT SHOULD HAVE"
017446  042111  047040  052117
```

```
 4015  017454  052040  040522  020120
 4016  017462  047117  043040  050120
 4017  017470  042440  041530  050105
 4018  017476  044524  047117  020054
 4019  017504  052502  020124  052111
 4020  017512  051440  047510  046125
 4021  017520  020104  040510  042526
 4022  017526     000
 4023  017527     103  052520  052040  EMQ:    .ASCIZ  "CPU TRAPPED ON FPP EXCEPTION, BUT IT SHOULD NOT HAVE"
 4024  017534  040522  050120  042105
 4025  017542  047440  020116  050106
 4026  017550  020120  054105  042503
 4027  017556  052120  047511  026116
 4028  017564  041040  052125  044440
 4029  017572  020124  044123  052517
 4030  017600  042114  047040  052117
 4031  017606  044040  053101  000105
 4032  017614  043101  042524  020122  EMR:    .ASCIZ  "AFTER TRAP, STACK POINTER IS INCORRECT"
 4033  017622  051124  050101  020054
 4034  017630  052123  041501  020113
 4035  017636  047520  047111  042524
 4036  017644  020122  051511  044440
 4037  017652  041516  051117  042522
 4038  017660  052103     000
 4039  017663     101  052106  051105  EMS:    .ASCIZ  "AFTER TRAP, STORED PC IS INCORRECT"
 4040  017670  052040  040522  026120
 4041  017676  051440  047524  042522
 4042  017704  020104  041520  044440
 4043  017712  020123  047111  047503
 4044  017720  051122  041505  000124
 4045  017726  043101  042524  020122  EMT:    .ASCIZ  "AFTER TRAP, STORED PS IS INCORRECT"
 4046  017734  051124  050101  020054
 4047  017742  052123  051117  042105
 4048  017750  050040  020123  051511
 4049  017756  044440  041516  051117
 4050  017764  042522  052103     000
 4051  017771     101  052106  051105  EMU:    .ASCIZ  "AFTER TRAP, LOADED PS IS INCORRECT"
 4052  017776  052040  040522  026120
 4053  020004  046040  040517  042504
 4054  020012  020104  051520  044440
 4055  020020  020123  047111  047503
 4056  020026  051122  041505  000124
 4057
 4058
 4059
 4060
 4061                                  ;DATA HEADERS HERE
 4062  020034  043040  051520     000  DHA:    .ASCIZ  " FPS"
 4063  020041     040  042506  000103  DHB:    .ASCIZ  " FEC"
 4064  020046  043040  040505     000  DHC:    .ASCIZ  " FEA"
 4065  020053     040  050106  004523  DHD:    .ASCIZ  " FPS     R0"
 4066  020060  020040  030122     000
 4067  020065     040  051040  000060  DHE:    .ASCIZ  "  R0"
 4068  020072  020040  030522  020011  DHF:    .ASCIZ  "  R1     SP"
 4069  020100  051440  000120
 4070  020104  051044  043505  000060  DHG:    .ASCIZ  "$REG0"
```

```
4071  020112  051044  043505  004460  DHH:    .ASCIZ  "$REG0  $REG1"
4072  020120  051044  043505  000061
4073  020126  051044  043505  004460  DHI:    .ASCIZ  "$REG0  $REG1   $REG2   $REG3"
4074  020134  051044  043505  004461
4075  020142  051044  043505  004462
4076  020150  051044  043505  000063
4077  020156  054105  023520  026504  DHJ:    .ASCIZ  "EXP'D-FPS-RCV'D"
4078  020164  050106  026523  041522
4079  020172  023526  000104
4080  020176  054105  023520  026504  DHK:    .ASCIZ  "EXP'D-FEC-RCV'D        EXP'D-FEA-RCV'D"
4081  020204  042506  026503  041522
4082  020212  023526  004504  054105
4083  020220  023520  026504  042506
4084  020226  026501  041522  023526
4085  020234  000104
4086  020236  054105  023520  004504  DHL:    .ASCIZ  "EXP'D  RCV'D"
4087  020244  041522  023526  000104
4088  020252  043040  051520  020011  DHM:    .ASCIZ  " FPS    FEC     FEA    OLD PS  OLD PC  SP AFTER"
4089  020260  042506  004503  043040
4090  020266  040505  047411  042114
4091  020274  050040  004523  046117
4092  020302  020104  041520  051411
4093  020310  020120  043101  042524
4094  020316  000122
4095
4096
4097                                  ;DATA VECTORS HERE
4098                                  .EVEN
4099  020320  001466  000000        DTB:    .WORD   FEC,0
4100  020324  001464                DTD:    .WORD   FPS
4101  020326  001526  000000        DTE:    .WORD   EREG0,0
4102  020332  001530  001542  000000 DTF:   .WORD   EREG1,EREG6,0
4103  020340  001170  000000        DTG:    .WORD   $REG0,0
4104  020344  001170  001172  000000 DTH:   .WORD   $REG0,$REG1,0
4105  020352  001170  001172  001174 DTI:   .WORD   $REG0,$REG1,$REG2,$REG3,0
4106  020360  001176  000000
4107  020364  001220                DTJ:    .WORD   $TMP4
4108  020366  001464  000000        DTA:    .WORD   FPS,0
4109  020372  001222  001466  001500 DTK:   .WORD   $TMP5,FEC,EXPFEA
4110  020400  001470  000000        DTC:    .WORD   FEA,0
4111  020404  001464  001466  001470 DTL:   .WORD   FPS,FEC,FEA,FPPOPS,FPPOPC,OSPRCV,0
4112  020412  001474  001472  001514
4113  020420  000000
4114  020422  001504  001514  000000 DTAG:  .WORD   EXPSP,OSPRCV,0
4115  020430  001502  001506  000000 DTAH:  .WORD   EXPRTS,OPCRCV,0
4116  020436  001214  001510  000000 DTAI:  .WORD   $TMP2,OPSRCV,0
4117  020444  001212  001512  000000 DTAJ:  .WORD   $TMP1,NPSLOD,0
4118
4119                                  ;THE END
4120          000001                  .END
```

```
AAD     012630          3020#   3022
AADDM   013114          3088#   3090
ABASE = 000000          1027
ACDW1 = 000000          1027
ACDW2 = 000000          1027
ACPUOP= 000000          1027    1042
ADDW0 = 000000          1027
ADDW1 = 000000          1027
ADDW10= 000000          1027
ADDW11= 000000          1027
ADDW12= 000000          1027
ADDW13= 000000          1027
ADDW14= 000000          1027
ADDW15= 000000          1027
ADDW2 = 000000          1027
ADDW3 = 000000          1027
ADDW4 = 000000          1027
ADDW5 = 000000          1027
ADDW6 = 000000          1027
ADDW7 = 000000          1027
ADDW8 = 000000          1027
ADDW9 = 000000          1027
ADEVCT= 000000          1027    1033
ADEVM = 000000          1027
AD0101  001604          1135#   1510    2034
AD1000  001610          1137#   1546    1548
AD1001  001606          1136#
ADSTOI  001760          1164#
AENV  = 000000          1027    1038
AENVM = 000000          1027    1039
AFATAL= 000000          1027    1030
AMADR1= 000000          1027
AMADR2= 000000          1027
AMADR3= 000000          1027
AMADR4= 000000          1027
AMAMS1= 000000          1027
AMAMS2= 000000          1027
AMAMS3= 000000          1027
AMAMS4= 000000          1027
AMSGAD= 000000          1027    1035
AMSGLG= 000000          1027    1036
AMSGTY= 000000          1027    1029
AMTYP1= 000000          1027
AMTYP2= 000000          1027
AMTYP3= 000000          1027
AMTYP4= 000000          1027
APASS = 000000          1027    1032
APRIOR= 000000          1027
APTCSU= 000040          3692    3797#
APTENV= 000001          3572    3685    3753    3795#
APTSIZ= 000200          1231    3794#
APTSPO= 000100          3687    3755    3796#
ASCHOT  017054          1342    3962#
ASCWRM  017062          1344    3963#
ASWREG= 000000          1027    1040
ATESTN= 000000          1027    1031
```

```
AUNIT = 000000          1027    1034
AUSWR = 000000          1027    1041
AVECT1= 000000          1027
AVECT2= 000000          1027
ASREGO  001632          1142#   1589*   1941*   1954*   2210    2212
BGNMES  001772          1170#   1270
BIT0  = 000001           869#   1355    3448
BIT00 = 000001           859#    869
BIT01 = 000002           858#    868
BIT02 = 000004           857#    867
BIT03 = 000010           856#    866
BIT04 = 000020           855#    865    1276    1324    2683    2935    3221
BIT05 = 000040           854#    864
BIT06 = 000100           853#    863
BIT07 = 000200           852#    862
BIT08 = 000400           851#    861    3499
BIT09 = 001000           850#    860    3507    3582
BIT1  = 000002           868#
BIT10 = 002000           849#   3560
BIT11 = 004000           848#   3514
BIT12 = 010000           847#   1308    1334    1336    1339    2687    2939    3227
BIT13 = 020000           846#   3567
BIT14 = 040000           845#   3306    3320    3485
BIT15 = 100000           844#   1258    3422    3428
BIT2  = 000004           867#   3436
BIT3  = 000010           866#
BIT4  = 000020           865#
BIT5  = 000040           864#
BIT6  = 000100           863#   1350    1355    3423    3448
BIT7  = 000200           862#   1248
BIT8  = 000400           861#
BIT9  = 001000           860#
BPTVEC= 000014           876#
CLKDON  002536          1264    1267#
CLKPRS  001516          1110#   1241*   1248*   1258*   1348    1353    3191    3195
CR    = 000015           784#   3731    3741
CRLF  = 000200           785#   3702    3741
DALTA   001634          1143#   1947    1960
DALTB   001644          1145#   1948    2263    2277
DALTC   001654          1147#   1967
DBIG    001612          1138#   1624    1830    2513    2514
DDISP = 177570           791#    990    1219
DDIV    012736          3044#   3046
DHA     020034          1069    4062#
DHB     020041          1070    4063#
DHC     020046          1071    4064#
DHD     020053          1072    4065#
DHE     020065          1073    1079    4067#
DHF     020072          1074    4068#
DHG     020104          1078    4070#
DHH     020112          1077    4071#
DHI     020126          1076    4073#
DHJ     020156          1088    4077#
DHK     020176          1087    4080#
DHL     020236          1083    1084    1085    1086    4086#
DHM     020252          1090    4088#
```

```
DISPLA  001150            990#  1219*  1227*  3529*  3559*
DISPRE  000174            922#  1227
DMZERO  001622           1140#  1709   1809   2324   2345   2352   2465
DSMALL  001574           1132#
DSWR  = 177570            790#   989   1218
DTA     020366           1069   4108#
DTAG    020422           1083   4114#
DTAH    020430           1084   4115#
DTAI    020436           1085   4116#
DTAJ    020444           1086   4117#
DTB     020320           1070   4099#
DTC     020400           1071   4110#
DTD     020324           1072   4100#
DTE     020326           1073   1079   4101#
DTF     020332           1074   4102#
DTG     020340           1078   4103#
DTH     020344           1077   4104#
DTI     020352           1076   4105#
DTJ     020364           1088   4107#
DTK     020372           1087   4109#
DTL     020404           1090   4111#
D0100   001724           1157#  2117
D0100X  001736           1160#  2107
D0101   001560           1130#  1135   1489   1497   1500   1503   1506   1865   2052   2145   3146
D0111   001744           1161#  2104   2112
D1000   001726           1137   1158#  2047   2099
D1001   001566           1131#  1136   2075
D1010   001556           1129#  1436   1438   1461   1464   1467   1470   1525   1532   1535   1676   1690   1693
                         1823   1840   2035   2037   2054   2056   2073   2082   2094   2146   2148   3135   3138
D20     001714           1155#  2283   2290
D37     001674           1151#  2303
D40     001664           1149#  2305   2323   2338
D46     001704           1153#  2302
DST01   001754           1163#  1164   2226   2250   2254
EMA     017071           1069   1088   3967#
EMB     017115           1070   3971#
EMC     017141           1071   3975#
EMD     017165           1072   3979#
EME     017217           1073   3984#
EMF     017241           1074   3988#
EMH     017263           1076   1077   1078   1079   3992#
EMI     017335           1087   4000#
EMJ     017365           1090   4005#
EMP     017441           1075   1081   4013#
EMQ     017527           1082   4023#
EMR     017614           1083   4032#
EMS     017663           1084   4039#
EMT     017726           1085   4045#
EMTVEC= 000030            879#  1202*  1203*
EMU     017771           1086   4051#
EMV001  001274           1069#
EMV002  001302           1070#
EMV003  001310           1071#
EMV004  001316           1072#
EMV005  001324           1073#
EMV006  001332           1074#
```

```
EMV007  001340          1075#
EMV010  001346          1076#
EMV011  001354          1077#
EMV012  001362          1078#
EMV013  001370          1079#
EMV014  001376          1081#
EMV015  001404          1082#
EMV016  001412          1083#
EMV017  001420          1084#
EMV020  001426          1085#
EMV021  001434          1086#
EMV022  001442          1087#
EMV023  001450          1088#
EMV024  001456          1090#
EREG0   001526          1117#   3548#   4101
EREG1   001530          1118#   3549#   4102
EREG2   001532          1119#   3550#
EREG3   001534          1120#   3551#
EREG4   001536          1121#   3552#
EREG5   001540          1122#   3553#
EREG6   001542          1123#   3554#   3555#   4102
EREG7   001544          1124#   3556#
ERRVEC= 000004           872#   1216    1217#   1228#   1242    1243    1244#   1245#   1255#   1267#   1268#   3490    3491#
                        3493#   3496#
EXLOP   013246          3124#   3129
EXPFEA  001500          1102#   2566#   2585#   2604#   2623#   2642#   2661#   2694#   2723#   2742#   2761#   2780#   2799#
                        2818#   2837#   2856#   2875#   2894#   2913#   2946#   3354#   3357    4109
EXPRTS  001502          1103#   2567#   2586#   2605#   2624#   2643#   2662#   2695#   2724#   2743#   2762#   2781#   2800#
                        2819#   2838#   2857#   2876#   2895#   2914#   2947#   3331    4115
EXPSP   001504          1104#   3324#   3325#   3326    4114
EX4     013304          3135#
FCONT1  014200          3307    3310    3346#
FEA     001470          1098#   2333    2380    2406    2431    2455    2479    2503    2529    2555    3357    4110    4111
FEC     001466          1097#   2326#   2330    2373#   2377    2399#   2403    2424#   2428    2448#   2452    2472#   2476
                        2496#   2500    2522#   2526    2548#   2552    3107#   3111    3347#   3355    3456#   4099    4109
                        4111
FEC02   014300          2564    2721    3372#
FEC04   014310          2583    2740    3377#
FEC06   014326          2602    2759    2778    3382#
FEC10   014344          2621    2797    2816    3387#
FEC12   014364          2640    2835    2854    3393#
FEC14   014402          2659    2873    2892    3398#
FEC16   014414          2692    2911    2944    3403#
FII02   014300          2566    2723    3373#
FII04   014314          2585    2742    3378#
FII06   014332          2604    2761    2780    3383#
FII10   014354          2623    2799    2818    3389#
FII12   014370          2642    2837    2856    3394#
FII14   014402          2661    2875    2894    3399#
FII16   014422          2694    2913    2946    3405#
FLPERR  014016          3290    3293#
FLTMZR  014436          3399    3411#
FLTONE  001546          1127#   3171
FLTZER  014432          3378    3410#
FLT001  014442          3377    3382    3387    3412#
FLT002  014446          3388    3413#
```

```
FLT003  014452          3393    3415#
FLT004  014456          3394    3416#
FPPILT  014620          1237    3452#
FPPNTP  014036          3306#   3375    3380    3385    3391    3396    3401    3407
FPPN1   011002          2565    2574#
FPPN12  011576          2760    2769#
FPPN14  011716          2798    2807#
FPPN16  012036          2836    2845#
FPPN2   011052          2584    2593#
FPPN20  012156          2874    2883#
FPPN22  012276          2912    2921#
FPPN3   011122          2603    2612#
FPPN4   011172          2622    2631#
FPPN5   011242          2641    2650#
FPPN6   011312          2660    2669#
FPPN7   011406          2693    2702#
FPPOPC  001472          1099#   3453*   3467    4111
FPPOPS  001474          1100#   3454*   3466    4111
FPPOSP  001476          1101#
FPPTRP  014052          3283    3313#
FPPT10  011456          2722    2731#
FPPT11  011526          2741    2750#
FPPT13  011646          2779    2788#
FPPT15  011766          2817    2826#
FPPT17  012106          2855    2864#
FPPT21  012226          2893    2902#
FPPT23  012372          2945    2954#
FPPVEC= 000244           903#   1237*   1238*   3281    3282    3283*   3284*   3364*   3365*
FPS     001464          1096#   1370*   1371    1376*   1377    1382*   1383    1388*   1389    1394*   1395    1400*   1401
                        1450*   1451    1480*   1490*   1491    1515*   1552*   1570*   1577*   1584*   1591*   1597*   1603*
                        1613*   1614    1619*   1620    1625*   1626    1636*   1637    1642*   1643    1648*   1649    1656*
                        1657    1667*   1668    1680*   1681    1699*   1700    1711*   1712    1727*   1728    1739*   1740
                        1755*   1756    1769*   1770    1788*   1789    1802*   1811*   1812    1825*   1826    1842*   1843
                        1867*   1868    1905*   1906    1917*   1918    1936*   1937    1949*   1950    1961*   1962    1978*
                        1979    1990*   1991    2007*   2008    2019*   2020    2041*   2042    2060*   2061    2076*   2084*
                        2128*   2129    2152*   2153    2171*   2183*   2198*   2199    2216*   2227*   2228    2244*   2245
                        2249*   2265*   2266    2284*   2285    2325*   2327    2339*   2346*   2347    2353*   2367*   2372*
                        2374    2393*   2398*   2400    2418*   2423*   2425    2442*   2447*   2449    2466*   2471*   2473
                        2490*   2495*   2497    2515*   2521*   2523    2542*   2547*   2549    2970*   2985*   3006*   3024*
                        3038*   3050*   3063*   3074*   3092*   3106*   3131*   3141*   3153*   3164*   3173*   3346*   3349
                        3455*   3457    3460    4100    4108    4111
F5T01   001762          1165#   2172    2233
GNS     = ****** U       921    3906    3907    3908    3909
HOTWRM  015426          1342*   1344*   3610#
HT      = 000011         782#   3700    3741
IKW11L  014462          1250    3420#
IKW11P  014510          1261    3426#
IOTVEC= 000020           877#   1200*   1201*
KW11LC  001520          1112#   1249*   3421*   3422*
KW11PC  001522          1113#   1259*   3427*   3428*
KW11PR  001524          1114#   1260*   1263    3434*   3436    3442*   3443*   3445*   3447
LF      = 000012         783#   3735    3741
LKS     = 177546         894#   1247*   1350*   3193*   3420*   3423*
LKV     = 000100         895#   1250*   1251*
MED     = 076600         886#   1275    1323    1327    1337    2682    2686    2934    2938    3220    3228    3234    3938
MMUL    012672          3032#   3034
```

```
MORE    013404          3156#
NEWPAS  002720          1279    1281    1306#   3275
NKWL1   003076          1349    1353#
NKWL2   013546          3192    3195#
NKWP1   003112          1354    1357#
NKWP2   013560          3196    3198#
NKW11L  002450          1244    1253#
NKW11P  002532          1255    1265#
NPSLOD  001512          1107#   3313*   3340    4117
NWPAS1  002067          1181#   1311
NXT02   014302          2567    2724    3374#
NXT04   014320          2586    2743    3379#
NXT06   014336          2605    2762    2781    3384#
NXT10   014356          2624    2800    2819    3390#
NXT12   014374          2643    2838    2857    3395#
NXT14   014406          2662    2876    2895    3400#
NXT16   014424          2695    2914    2947    3406#
OPCRCV  001506          1105#   3314*   3331    4115
OPSRCV  001510          1106#   3315*   3335    4116
OSPRCV  001514          1108#   3316*   3326    3452*   4111    4114
PIRQ    = 177772         789#
PIRQVE= 000240           883#
PLKD    = 172544         899#
PLKR    = 172542         898#   1263*   3447*
PLKS    = 172540         897#   1257*   1355*   3197*   3426*   3448*
PLKV    = 000104         900#   1261*   1262*
PR0     = 000000         806#
PR1     = 000040         807#
PR2     = 000100         808#
PR3     = 000140         809#
PR4     = 000200         810#
PR5     = 000240         811#
PR6     = 000300         812#   1238    1251    1262
PR7     = 000340         813#   1245
PS      = 177776         786#    787    1410    1418    1426    1854    2177*   2179
PSW     = 177776         787#   1357*   3296*   3313    3367*
PWRVEC= 000024           878#   1206*   1207*   3916*   3917*   3926*   3932*   3946*   3947*
RESVEC= 000010           873#
RFLAG   = 000144         890#   1327    2686    2938    3228
R%HAMI= 000022           888#   1275    1323    2682    2934    3220
STACK   = 001100         777#   1198    1306
START   002100           925    1191#   3951
STKLMT= 177774           788#
SUBFM   013036          3069#   3071
SUBPAS  002756          1309    1323#   3238
SUBT    012550          3002#   3004
SWR     001146           989#   1196    1218*   1220    1226*   1233*   1308    1329    1332    3224    3262    3485    3499
                        3507    3514    3560    3567    3579    3582    3924    3939*
SWREG   000176           923#   1226
SW0     = 000001         841#
SW00    = 000001         831#    841    1332
SW01    = 000002         830#    840    1329    3224
SW02    = 000004         829#    839
SW03    = 000010         828#    838
SW04    = 000020         827#    837
SW05    = 000040         826#    836
```

```
SW06   = 000100            825#    835
SW07   = 000200            824#    834
SW08   = 000400            823#    833
SW09   = 001000            822#    832
SW1    = 000002            840#
SW10   = 002000            821#   3262
SW11   = 004000            820#
SW12   = 010000            819#
SW13   = 020000            818#
SW14   = 040000            817#
SW15   = 100000            816#
SW2    = 000004            839#
SW3    = 000010            838#
SW4    = 000020            837#
SW5    = 000040            836#
SW6    = 000100            835#
SW7    = 000200            834#
SW8    = 000400            833#
SW9    = 001000            832#
TBITVE= 000014             874#
TKVEC  = 000060            881#
TKW11P  002454            1252    1255#
TPVEC  = 000064            882#
TRAPVE= 000034             880#   1204*   1205*
TRPTST  013744            2569    2588    2607    2626    2645    2664    2697    2726    2745    2764    2783    2802    2821
                          2840    2859    2878    2897    2916    2949    3281#
TRTVEC= 000014             875#
TST1    003116            1368#
TST10   004326            1633#
TST11   004476            1674#
TST12   004674            1723#
TST13   005112            1783#
TST14   005324            1838#
TST15   005560            1901#
TST16   005672            1932#
TST17   006050            1974#
TST2    003264            1407#
TST20   006162            2003#
TST21   006274            2032#
TST22   006526            2092#
TST23   006642            2125#
TST24   007110            2192#
TST25   007354            2261#
TST26   007566            2321#
TST27   007744            2361#
TST3    003370            1434#
TST30   010050            2387#
TST31   010154            2413#
TST32   010254            2438#
TST33   010350            2462#
TST34   010444            2486#
TST35   010536            2510#
TST36   010642            2536#
TST37   010746            2563#
TST4    003566            1487#
TST40   011016            2572    2582#
```

```
TST41   011066              2591    2601#
TST42   011136              2610    2620#
TST43   011206              2629    2639#
TST44   011256              2648    2658#
TST45   011326              2667    2677#
TST46   011422              2700    2720#
TST47   011472              2729    2739#
TST5    003716              1522#
TST50   011542              2748    2758#
TST51   011612              2767    2777#
TST52   011662              2786    2796#
TST53   011732              2805    2815#
TST54   012002              2824    2834#
TST55   012052              2843    2853#
TST56   012122              2862    2872#
TST57   012172              2881    2891#
TST6    004042              1559#
TST60   012242              2900    2910#
TST61   012312              2919    2929#
TST62   012406              2952    2962#
TST63   012452              2977#
TST64   012512              2992#
TST65   012602              3013#
TST66   012766              3056#
TST67   013066              3081#
TST7    004236              1610#
TST70   013142              3100#
TST71   013222              3118#
TYPE  = 104401              1270    1278    1280    1311    1316    3264    3562    3570    3609    3628    3630    3631    3634
                            3636    3639    3642    3649    3653    3705    3858    3906#   3948
TYPOC = 104402              3618    3638    3641    3646    3907#
TYPON = 104404              3909#
TYPOS = 104403              1314    3908#
WEIRD   001574              1133#   2067
WFLAG = 000344               891#   1337    3234
$APTHD  001000               947     953#
$ASTAT= ****** U            3775    3790
$ATYC   016146              3746    3748#
$ATY1   016122              3744#
$ATY3   016130              3690    3745#
$ATY4   016140              3575    3747#
$AUTOB  001142               986#
$BDAOR  001130               981#
$BDOAT  001134               983#
$BELL   001234              1019#   3264    3562    3594
$CHARC  016116              3707*   3717*   3724    3733*   3738#
$CKSWR= ****** U            3912
$CMTAG  001100               968#   1193    1194    1202    1208    1209    1210
$CM1  = 000010              1001#   1002#   1003#   1004#   1005#   1006#   1007#   1008#   1009#
$CM2  = 000020              1001#   1002#   1003#   1004#   1005#   1006#   1007#   1008#   1009#
$CM3  = 000010               999#   1001
$CM4  = 000010              1009#   1010#   1011#   1012#   1013#   1014#   1015#   1016#   1017#
$CPUOP  001272              1042#
$CRLF   001241              1021#   1316    3570    3594    3610    3630    3636    3653    3706    3741
$DEVCT  001254              1033#
$DOAGN  013740              3258    3267    3275#
```

```
SENDAD  013730            933   3270#  3589
SENDCT  013700           1208   3260#
SENV    001264           1038#  3572   3685   3753   3777
SENVM   001265           1039#  1231   3687   3692   3755
SEOP    013642           3222   3225   3237   3250#
SEOPCT  013672           1208*  3257#  3261
SERFLG  001104            971#  3251#  3503   3505   3511*  3533   3557*  3594
SERMAX  001122            978#  1211#  3505   3528#  3533
SERROR  015162           1202   3547#
SERRPC  001124            979#  3564#  3565#  3566   3594   3617   3656
SERRTB  001274           1053#  3625
SERTTL  001116            976#  3563#  3594
SESCAP  001232           1018#  1210#  3527#  3585   3587   3594
SETABL  001264           1037#
SETEND  001274            959   1049#
SFATAL  001246           1030#  3781#
SFFLG   016366           3744#  3747#  3775   3784#  3792#
SFILLC  001164            997#  3710   3741
SFILLS  001163            996#  3741
SGDADR  001126            980#
SGDDAT  001132            982#
SGET42  013720           3263   3266#
SGTSWR= ******  U        3911
SHD     =  000000         755
SHIBTS  001000            954#
SICNT   001106            972#  3518#  3519   3521#  3532
SILLUP  017036           3916   3932   3953#
SINTAG  001143            987#
SITEMB  001120            977#  3566#  3574   3594   3614
SLF     001242           1022#  3594   3741
SLFLG   016365           3785#  3791#
SLPADR  001110            973#  1212#  3509#  3525#  3530   3532
SLPERR  001114            975#  1213#  3290#  3509   3526#  3532   3584
SLPTST  001112            974#  3501
SMAIL   001244            955    959   1028#  1230   3524   3572   3685
SMBADR  001002            955#
SMFLG   016364           3745#  3751   3786#  3790#
SMSGAD  001260           1035#  3761#  3764
SMSGLG  001262           1036#  3766#
SMSGTY  001244           1029#  3759   3767#  3779   3783#
SMXCNT  015160           3522   3532#
SNULL   001162            995#  3712   3741
SNWTST= 000001           1365#  1404#  1431#  1484#  1519#  1556#  1607#  1630#  1671#  1720#  1780#  1835#  1898#
                         1929#  1971#  2000#  2029#  2089#  2122#  2189#  2258#  2318#  2358#  2384#  2410#  2435#
                         2459#  2483#  2507#  2533#  2560#  2579#  2598#  2617#  2636#  2655#  2674#  2717#  2736#
                         2755#  2774#  2793#  2812#  2831#  2850#  2869#  2888#  2907#  2926#  2959#  2974#  2989#
                         3010#  3053#  3078#  3097#  3115#
SOCNT   016612           3830#  3859#  3872#
SOMODE  016614           3825#  3829#  3834   3837#  3848#  3874#
SOVER   015144           3486   3502   3510   3520   3529#
SPASS   001252           1032#  1230#  1312   3254#  3255#  3516   3533
SPASTM  001006            957#
SPOWER  017044           3949   3956#
SPWRAD  017032           3951#
SPWRDN  016664           1206   3916#  3946
SPWRMG  017026           3949#
```

```
$PWRUP  016736          3926    3932#
$QUES   001240          1020#   3594    3741
$RDCHR= ****** U        3912
$RDOEC= ****** U        3912
$RDLIN= ****** U        3912
$RDOCT= ****** U        3912
$REGAD  001166           999#
$REGO   001170          1001#   1142    1456    1495    1497    1526    1532    1542    1568*   1575*   1576    1581    1583
                        1590    1677*   1679*   1685    1703*   1715*   1732*   1738*   1744    1760*   1774*   1786*   1787*
                        1793    1803*   1816*   1829*   1847*   1853*   1860    1872*   1873    1881*   1882    1910*   1922*
                        1966*   1983*   1995*   2012*   2024*   2046*   2066*   2074*   2083*   2098*   2106*   2116*   2133*
                        2157*   2170*   2178*   2185    2203*   2217    2232*   2240    2250    2270*   2276*   2289*   2295*
                        2304*   2310*   2337*   2351*   2366*   2392*   2417*   2516*   2541*   2968*   2969    2983*   2984
                        3005*   3007    3017*   3025    3036*   3048*   3059    3062    3072*   3073    3091*   3105*   3108
                        3130*   3132    3136    3142    3151*   3152    3163*   3165    3174*   3175    3179*   3180    3383*
                        4103    4104    4105
$REG1   001172          1002#   1500    1535    1748    1797    1877    2085    2221    2254    4104    4105
$REG2   001174          1003#   1503    1528    1690    2238    4105
$REG3   001176          1004#   1506    1693    4105
$REG4   001200          1005#   1540    2993    2995*   3002    3023*   3140
$REG5   001202          1006#
$REG6   001204          1007#
$REG7   001206          1008#
$R2A    = ****** U      3912
$SAVRE= ****** U        3912
$SAVR6  017042          3925*   3933    3934*   3935*   3955#
$SCOPE  014704          1200    3483#
$SETUP= 000037          1184#   1199    1200    1202    1204    1206    1208    1209    1210    1212    3484    3548    3582
                        3589
$STUP = 177777          1184#
$SVLAD  015110          3494    3523#
$SVPC = 000204           931#    936
$SWR  = 167400           744#    755    1017    1018    1019    1209    1210    1212    1213    1369    1408    1435    1488
                        1523    1560    1611    1634    1675    1724    1784    1839    1902    1933    1975    2004    2033
                        2093    2126    2193    2262    2322    2362    2388    2414    2439    2463    2487    2511    2537
                        2564    2583    2602    2621    2640    2659    2678    2721    2740    2759    2778    2797    2816
                        2835    2854    2873    2892    2911    2930    2963    2978    2993    3014    3057    3082    3101
                        3119    3475    3476    3477    3478    3479    3485    3497    3499    3500    3503    3504    3505
                        3512    3513    3514    3526    3529    3532    3539    3540    3541    3542    3543    3560    3567
                        3579    3582    3594    3952
$SWREG  001266          1040#   1233
$SWRMK= 000000          3479    3501
$TESTN  001250          1031#   3524*   3655
$TIMES  001230          1017#   1209*   3253*   3512*   3519    3522*   3532
$TKB    001154           992#
$TKS    001152           991#
$TMP0   001210          1009#   3287
$TMP1   001212          1010#   4117
$TMP2   001214          1011#   4116
$TMP3   001216          1012#
$TMP4   001220          1013#   4107
$TMP5   001222          1014#   4109
$TMP6   001224          1015#
$TMP7   001226          1016#
$TN   = 000072           744#    755    1365    1369#   1404    1408#   1431    1435#   1484    1488#   1519    1523#   1556
                        1560#   1607    1611#   1630    1634#   1671    1675#   1720    1724#   1780    1784#   1835    1839#
```

```
                        1898    1902#   1929    1933#   1971    1975#   2000    2004#   2029    2033#   2089    2093#   2122
                        2126#   2189    2193#   2258    2262#   2318    2322#   2358    2362#   2384    2388#   2410    2414#
                        2435    2439#   2459    2463#   2483    2487#   2507    2511#   2533    2537#   2560    2564#   2572
                        2579    2583#   2591    2598    2602#   2610    2617    2621#   2629    2636    2640#   2648    2655
                        2659#   2667    2674    2678#   2700    2717    2721#   2729    2736    2740#   2748    2755    2759#
                        2767    2774    2778#   2786    2793    2797#   2805    2812    2816#   2824    2831    2835#   2843
                        2850    2854#   2862    2869    2873#   2881    2888    2892#   2900    2907    2911#   2919    2926
                        2930#   2952    2959    2963#   2974    2978#   2989    2993#   3010    3014#   3053    3057#   3078
                        3082#   3097    3101#   3115    3119#
$TPB    001160          994#    3730*   3741
$TPFLG  001165          998#    3679    3741
$TPS    001156          993#    3728    3741
$TRAP   016616          1204    3883#
$TRAP2  016640          3894#   3905
$TRP  = 000005          3898#   3907#   3908#   3909#   3910#
$TRPAD  016652          3888    3905#
$TSTM   001004          956#
$TSTNM  001102          970#    1345*   3252*   3501    3523*   3524    3529    3533    3559    3594
$TYPBN= ****** U        3910
$TYPDS= ****** U        3910
$TYPE   015640          3679#   3772    3898    3906
$TYPEC  016052          3709    3716    3723    3728#   3729
$TYPER  015424          3569    3608#
$TYPEX  016120          3734    3736    3739#
$TYPOC  016414          3828#   3907
$TYPON  016430          3827    3830#   3909
$TYPOS  016370          3823#   3908
$UNIT   001256          1034#
$UNITM  001010          958#
$USWR   001270          1041#
$XTSTR  014714          3488#
$OFILL  016613          3824*   3828*   3838    3873#
$40CAT= ****** U        3485    3569
.       020452          917#    921#    931     932#    934#    936#    937#    943     944#    946#    948#    967#    1023
                        1197    1212    1213    1372    1378    1384    1390    1396    1402    1413    1421    1429    1439
                        1447    1452    1462    1465    1468    1471    1478    1482    1492    1498    1501    1504    1507
                        1517    1529    1533    1536    1543    1549    1554    1572    1579    1586    1593    1599    1605
                        1615    1621    1627    1638    1644    1650    1654    1658    1663    1669    1682    1687    1691
                        1694    1701    1706    1713    1718    1729    1735    1741    1745    1749    1757    1763    1771
                        1777    1790    1794    1798    1806    1813    1819    1827    1832    1844    1850    1857    1861
                        1869    1874    1878    1883    1889    1895    1907    1913    1919    1925    1938    1944    1951
                        1957    1963    1969    1980    1986    1992    1998    2009    2015    2021    2027    2038    2043
                        2049    2057    2062    2069    2078    2086    2101    2109    2119    2130    2136    2141    2149
                        2154    2160    2165    2174    2181    2186    2195    2200    2207    2213    2218    2222    2229
                        2235    2241    2246    2251    2255    2267    2273    2279    2286    2292    2298    2307    2313
                        2328    2331    2334    2341    2348    2355    2375    2378    2381    2401    2404    2407    2426
                        2429    2432    2450    2453    2456    2474    2477    2480    2498    2501    2504    2524    2527
                        2530    2550    2553    2556    2971    2986    3008    3027    3039    3051    3065    3075    3095
                        3109    3112    3133    3144    3154    3166    3176    3181    3289    3374    3379    3384    3390
                        3395    3400    3406    3532    3533    3594    3661#   3741    3793#   3928    3954
.$ASTA= ****** U        3745    3748
.$X   = 001000          943#    948
```

```
CMPFLT    744#
COMMEN    884#
COMM00    744#
COMM01    744#
COMM02    744#
COMM03    744#
COMM04    744#
COMM05    744#
COMM06    744#
COMM07    744#
COMM1     744#
COMM10    744#
COMM11    744#
COMM12    744#
COMM13    744#
COMM14    744#
COMM15    744#
COMM16    744#
COMM17    744#
COMM2     744#
COMM20    744#
COMM21    744#
COMM22    744#
COMM23    744#
COMM24    744#
COMM25    744#
COMM26    744#
COMM27    744#
COMM3     744#
COMM30    744#
COMM31    744#
COMM32    744#
COMM33    744#
COMM34    744#
COMM35    744#
COMM36    744#
COMM37    744#
COMM4     744#
COMM40    744#
COMM41    744#
COMM42    744#
COMM43    744#
COMM44    744#
COMM45    744#
COMM46    744#
COMM47    744#
ENDCOM    884#
ERRCMP    744#  3326  3331  3335  3340
ERRLUR    744#  3593
ERROR     778#  1373  1379  1385  1391  1397  1403  1414  1422  1430  1440  1448  1453  1463  1466
              1469  1472  1479  1483  1493  1499  1502  1505  1508  1518  1530  1534  1537  1544  1550
              1555  1573  1580  1587  1594  1600  1606  1616  1622  1628  1639  1645  1651  1655  1659
              1664  1665  1666  1670  1683  1688  1692  1695  1702  1707  1714  1719  1730  1736  1742
              1746  1750  1758  1764  1772  1778  1791  1795  1799  1807  1814  1820  1828  1833  1845
              1851  1858  1862  1870  1875  1879  1884  1890  1896  1908  1914  1920  1926  1939  1945
              1952  1958  1964  1970  1981  1987  1993  1999  2010  2016  2022  2028  2039  2044  2050
```

```
                2058    2063    2070    2079    2087    2102    2110    2120    2131    2137    2142    2150    2155    2161    2166
                2175    2182    2187    2196    2201    2208    2214    2219    2223    2230    2236    2242    2247    2252    2256
                2268    2274    2280    2287    2293    2299    2308    2314    2329    2332    2335    2342    2349    2356    2370
                2376    2379    2382    2396    2402    2405    2408    2421    2427    2430    2433    2445    2451    2454    2457
                2469    2475    2478    2481    2493    2499    2502    2505    2519    2525    2528    2531    2545    2551    2554
                2557    2972    2987    3009    3028    3040    3052    3066    3076    3096    3110    3113    3134    3145    3155
                3167    3177    3182    3308    3322    3328    3333    3337    3342    3351    3359    3465
ESCAPE           884#
FCOM0            744#
FCOM1            744#
FCOM2            744#
FCOM3            744#
FCOM4            744#
FPRGT0           744#
FPRGT1           744#
FPSFEC           744#    3349
FPSTST           744#
GENCOM           744#
GENTS1           744#
GENTS2           744#
GENTS3           744#
GENTS4           744#    2560    2579    2598    2617    2636    2655    2674    2717    2736    2755    2774    2793    2812    2831
                2850    2869    2888    2907    2926
GETPRI           884#
GETSWR           884#
GTSTD            744#
GTSTF            744#
HTSTD            744#
HTSTF            744#
MOVDIS           744#    3285
MOVDI1          3285#
MOVDI2          3285#
MULT             884#
NEWTST           884#    1365    1404    1431    1484    1519    1556    1607    1630    1671    1720    1780    1835    1898    1929
                1971    2000    2029    2089    2122    2189    2258    2318    2358    2384    2410    2435    2459    2483    2507
                2533    2560    2579    2598    2617    2636    2655    2674    2717    2736    2755    2774    2793    2812    2831
                2850    2869    2888    2907    2926    2959    2974    2989    3010    3053    3078    3097    3115
POP              884#    3787    3788    3939    3940
PUSH             884#    3748    3750    3771    3918    3924
REPORT           884#
SBTST1           744#
SBTST2           744#
SCOM0            744#
SCOM1            744#
SCOM2            744#
SCOM3            744#
SCOM4            744#
SCOM5            744#
SCOM6            744#
SCOM7            744#
SCOPE            779#    1368    1407    1434    1487    1522    1559    1610    1633    1674    1723    1783    1838    1901    1932
                1974    2003    2032    2092    2125    2192    2261    2321    2361    2387    2413    2438    2462    2486    2510
                2536    2563    2582    2601    2620    2639    2658    2677    2720    2739    2758    2777    2796    2815    2834
                2853    2872    2891    2910    2929    2962    2977    2992    3013    3056    3081    3100    3118    3208
SCPLUR           744#    3484
SEADAT           744#
```

```
SETPRI    884#
SETREG    744#  3548
SETTRA   3898#  3907  3908  3909
SETUP     744#   884#  1191
SKIP      884#  2571  2590  2609  2628  2647  2666  2699  2728  2747  2766  2785  2804  2823  2842
         2861  2880  2899  2918  2951
SLASH     884#  1272  1299  2679  2690  2708  2715  2931  2942
SPACE     884#
STARS     884#   929   940   942   949   963  1023  1026  1187  1189  1302  1304  1319  1321  1361
         1363  1365  1367  1404  1406  1431  1433  1484  1486  1519  1521  1556  1558  1607  1609
         1630  1632  1671  1673  1720  1722  1780  1782  1835  1837  1898  1900  1929  1931  1971
         1973  2000  2002  2029  2031  2089  2091  2122  2124  2189  2191  2258  2260  2318  2320
         2358  2360  2384  2386  2410  2412  2435  2437  2459  2461  2483  2485  2507  2509  2533
         2535  2560  2562  2579  2581  2598  2600  2617  2619  2636  2638  2655  2657  2674  2676
         2717  2719  2736  2738  2755  2757  2774  2776  2793  2795  2812  2814  2831  2833  2850
         2852  2869  2871  2888  2890  2907  2909  2926  2928  2959  2961  2974  2976  2989  2991
         3010  3012  3053  3055  3078  3080  3097  3099  3115  3117  3201  3203  3205  3240  3472
         3535  3594  3664  3743  3800  3877  3914  3930
STATUS    744#  1376  1382  1388  1394  1400  1450  1490  1613  1619  1680  1699  1711  1727  1739
         1755  1769  1788  1811  1825  1842  1867  1905  1917  1936  1949  1961  1978  1990  2007
         2019  2041  2060  2128  2152  2198  2227  2244  2265  2284  2325  2346  2371  2397  2422
         2446  2470  2494  2520  2546
SWRSU     884#  1214#
TADDD1    744#
TADDD2    744#
TADDF1    744#
TADDF2    744#
TADDR1    744#
TADDR2    744#
TRMTRP   3898#
TYPBIN    884#
TYPDEC    884#
TYPNAM    884#
TYPNUM    884#
TYPOCS    884#
TYPOCT    884#
TYPTXT    884#
UPCODE    744#  3937
$$CMRE    961#  1001  1002  1003  1004  1005  1006  1007  1008
$$CMTM    961#  1009  1010  1011  1012  1013  1014  1015  1016
$$ESCA    884#
$$NEWT    884#  1365  1404  1431  1484  1519  1556  1607  1630  1671  1720  1780  1835  1898  1929
         1971  2000  2029  2089  2122  2189  2258  2318  2358  2384  2410  2435  2459  2483  2507
         2533  2560  2579  2598  2617  2636  2655  2674  2717  2736  2755  2774  2793  2812  2831
         2850  2869  2888  2907  2926  2959  2974  2989  3010  3053  3078  3097  3115
$$SET    3898#  3907  3908  3909
$$SETM   1230#
$$SKIP    884#  2572  2591  2610  2629  2648  2667  2700  2729  2748  2767  2786  2805  2824  2843
         2862  2881  2900  2919  2952
.EQUAT    744#   774
.HEADE    744#   745
.SBPAS    744#  3205
.SETUP    744#  1184
.STPAS    744#  1272
.$ACT1    744#   927
.$APTB    744#  1024#
```

```
.$APTH    744#    938
.$APTY    744#   3741
.$CATC    744#    915
.$CMTA    744#    961
.$EOP     744#   3240
.$ERRO    744#   3533
.$POWE    744#   3912
.$SCOP    744#   3470
.$TRAP    744#   3875
.$TYER    744#   3594
.$TYPE    744#   3662
.$TYPO    744#   3798

. ABS.  020452     000

 ERRORS DETECTED:  0
 DEFAULT GLOBALS GENERATED:  0

 DSKZ:DQFPCA,DSKZ:DQFPCA.SEQ/SOL/CRF=DQFPCA.MEM,DQFPCA.MAC,DQFPCA.P11
 RUN-TIME: 20 16 1 SECONDS
 RUN-TIME RATIO: 122/38=3.1
 CORE USED:  31K  (62 PAGES)
```